

日 本 国 特 許 庁  
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出 願 年 月 日  
Date of Application: 2 0 0 3 年 8 月 2 8 日

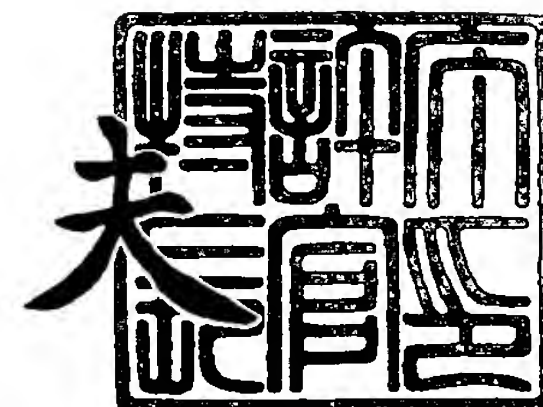
出 願 番 号  
Application Number: 特 願 2 0 0 3 - 3 0 5 5 1 1  
[ST. 10/C]: [ J P 2 0 0 3 - 3 0 5 5 1 1 ]

出 願 人  
Applicant(s): 株式会社リコー

2 0 0 3 年 9 月 3 0 日

特許庁長官  
Commissioner,  
Japan Patent Office

今 井 康 夫



【書類名】 特許願  
【整理番号】 0306365  
【提出日】 平成15年 8月28日  
【あて先】 特許庁長官 殿  
【国際特許分類】 G03G 21/00 396  
【発明者】  
    【住所又は居所】 東京都大田区中馬込 1 丁目 3 番 6 号 株式会社リコー内  
    【氏名】 松島 弘幸  
【特許出願人】  
    【識別番号】 000006747  
    【住所又は居所】 東京都大田区中馬込 1 丁目 3 番 6 号  
    【氏名又は名称】 株式会社リコー  
    【代表者】 桜井 正光  
【代理人】  
    【識別番号】 100080931  
    【住所又は居所】 東京都豊島区東池袋 1 丁目 2 0 番 2 号 池袋ホワイトハウスビル  
                    8 1 8 号  
    【弁理士】  
    【氏名又は名称】 大澤 敬  
【先の出願に基づく優先権主張】  
    【出願番号】 特願2002-276451  
    【出願日】 平成14年 9月24日  
【先の出願に基づく優先権主張】  
    【出願番号】 特願2002-272978  
    【出願日】 平成14年 9月19日  
【手数料の表示】  
    【予納台帳番号】 014498  
    【納付金額】 21,000円  
【提出物件の目録】  
    【物件名】 特許請求の範囲 1  
    【物件名】 明細書 1  
    【物件名】 図面 1  
    【物件名】 要約書 1  
    【包括委任状番号】 9809113

**【書類名】 特許請求の範囲****【請求項 1】**

通信クライアントから通信要求を受信し、前記通信クライアントに対してその通信要求に対する通信応答を送信し、前記通信クライアントからの動作要求であるクライアント要求を記載した通信要求を前記通信クライアントから受信し、該通信クライアントに、そのクライアント要求に対する動作応答を前記通信応答に記載して送信する通信サーバにおいて、

前記クライアント要求と前記通信クライアントに送信した動作要求であるサーバ要求に対する動作応答とを前記通信要求に記載した状態で一括して前記通信クライアントから受信する受信手段と、

その通信要求に対する通信応答として、前記通信クライアントから受信したクライアント要求に対する動作応答と前記サーバ要求とを一括して前記通信クライアントに送信する送信手段と、

前記クライアント要求に係る動作を実行し、実行結果としてそのクライアント要求に対する動作応答を生成する手段と設けたことを特徴とする通信サーバ。

**【請求項 2】**

請求項 1 記載の通信サーバであって、

前記動作要求は関数呼び出しであり、

前記動作応答はその関数呼び出しによって呼び出された関数の実行結果であることを特徴とする通信サーバ。

**【請求項 3】**

通信クライアントから通信要求を受信し、前記通信クライアントに対してその通信要求に対する通信応答を送信し、前記通信クライアントからの動作要求であるクライアント要求を記載した通信要求を前記通信クライアントから受信し、該通信クライアントに、そのクライアント要求に対する動作応答を前記通信応答に記載して送信する通信サーバにおいて、

前記クライアント要求とこの要求に対する動作応答とを記憶する第 1 の記憶手段と、

前記通信クライアントに対する動作要求であるサーバ要求とこの要求に対する動作応答とを記憶する第 2 の記憶手段と、

前記サーバ要求を生成して前記第 2 の記憶手段に記憶させる要求生成手段と、

前記第 1 の記憶手段からクライアント要求を読み出し、そのクライアント要求に係る動作を実行し、その実行結果としてそのクライアント要求に対する動作応答を生成し、その動作応答を読み出したクライアント要求と関連付けて前記第 1 の記憶手段に記憶させる応答生成手段と、

前記クライアント要求と前記通信クライアントに送信したサーバ要求に対する動作応答とを前記通信要求に記載した状態で一括して前記通信クライアントから受信する受信手段と、

該受信手段が受信したクライアント要求を前記第 1 の記憶手段に記憶させると共に、前記受信手段が受信した、前記通信クライアントに送信したサーバ要求に対する動作応答を、前記通信クライアントに送信したサーバ要求と関連付けて前記第 2 の記憶手段に記憶させる分配手段と、

前記クライアント要求に対する動作応答を前記第 1 の記憶手段から読み出すと共に、前記サーバ要求を前記第 2 の記憶手段から読み出す収集手段と、

前記受信手段が受信した通信要求に対する通信応答として、前記収集手段が読み出した動作応答とサーバ要求とを一括して前記通信クライアントに送信する送信手段とを設けたことを特徴とする通信サーバ。

**【請求項 4】**

請求項 3 記載の通信サーバであって、

前記受信手段が、前記通信クライアントから受信する動作応答と動作要求とを、それぞれ SOAP メッセージとして受信するようにし、

前記送信手段が、前記通信クライアントに送信すべき動作応答と動作要求とを、それぞれ SOAP メッセージとして送信するようにしたことを特徴とする通信サーバ。

【請求項 5】

請求項 3 又は 4 記載の通信サーバであって、

前記第 1 の記憶手段に記憶させたクライアント要求及び前記第 2 の記憶手段に記憶させたサーバ要求に優先順位を付す手段を設け、

前記応答生成手段を、前記優先順位の高いクライアント要求から順に読み出し、そのクライアント要求に対する動作応答を生成して前記第 1 の記憶手段に記憶させる手段とし、

前記収集手段を、前記優先順位の高いクライアント要求に対する動作応答から順に前記第 1 の記憶手段から読み出し、前記優先順位の高いサーバ要求から順に前記第 2 の記憶手段から読み出す手段としたことを特徴とする通信サーバ。

【請求項 6】

通信クライアントから HTTP リクエストを受信し、前記通信クライアントに対してその HTTP リクエストに対する HTTP レスポンスを送信し、前記通信クライアントからの SOAP リクエストを HTTP リクエストに記載した状態で前記通信クライアントから受信し、該通信クライアントにその SOAP リクエストに対する SOAP レスポンスを前記 HTTP レスポンスに記載して送信する通信サーバにおいて、

前記通信クライアントからの SOAP リクエストと前記通信クライアントに送信した SOAP リクエストに対する SOAP レスポンスとを 1 つの HTTP リクエストに記載した状態で前記通信クライアントから受信する受信手段と、

その HTTP リクエストに対する HTTP レスポンスとして、前記通信クライアントからの SOAP リクエストに対する SOAP レスポンスと前記通信クライアントに対する SOAP リクエストとを、1 つの HTTP レスポンスに記載して前記通信クライアントに送信する送信手段と、

前記通信クライアントから受信した SOAP リクエストに係る動作を実行し、その SOAP リクエストに対する SOAP レスポンスに記載すべき実行結果を生成する手段と設けたことを特徴とする通信サーバ。

【請求項 7】

請求項 6 記載の通信サーバであって、

前記 SOAP リクエストには関数呼び出しを記載し、

前記 SOAP レスポンスにはその関数呼び出しによって呼び出された関数の実行結果を記載するようにしたことを特徴とする通信サーバ。

【請求項 8】

通信クライアントから HTTP リクエストを受信し、前記通信クライアントに対してその HTTP リクエストに対する HTTP レスポンスを送信し、前記通信クライアントからの SOAP リクエストを HTTP リクエストに記載した状態で前記通信クライアントから受信し、該通信クライアントにその SOAP リクエストに対する SOAP レスポンスを前記 HTTP レスポンスに記載して送信する通信サーバにおいて、

前記通信クライアントからの動作要求であるクライアント要求とこの要求に対する動作応答とを記憶する第 1 の記憶手段と、

前記通信クライアントに対する動作要求であるサーバ要求とこの要求に対する動作応答とを記憶する第 2 の記憶手段と、

前記サーバ要求を生成して前記第 2 の記憶手段に記憶させる要求生成手段と、

前記第 1 の記憶手段からクライアント要求を読み出し、そのクライアント要求に係る動作を実行し、その実行結果としてそのクライアント要求に対する動作応答を生成し、その動作応答を読み出したクライアント要求と関連付けて前記第 1 の記憶手段に記憶させる応答生成手段と、

前記クライアント要求を記載した SOAP リクエストと、前記通信クライアントに送信した SOAP リクエストに対する SOAP レスポンスであって前記通信クライアントに送信したサーバ要求に対する動作応答を記載した SOAP レスポンスとを、1 つの HTTP

リクエストに記載した状態で前記通信クライアントから受信する受信手段と、

該受信手段が受信した S O A P リクエストに記載されたクライアント要求の内容を前記第 1 の記憶手段に記憶させると共に、前記受信手段が受信した S O A P レスポンスに記載された動作応答の内容を、前記通信クライアントに送信したサーバ要求と関連付けて前記第 2 の記憶手段に記憶させる分配手段と、

前記クライアント要求に対する動作応答を前記第 1 の記憶手段から読み出すと共に、前記サーバ要求を前記第 2 の記憶手段から読み出す収集手段と、

前記 1 つの H T T P リクエストに対する H T T P レスポンスとして、前記収集手段が読み出した動作応答の内容を記載した S O A P レスポンスと、前記収集手段が読み出した動作要求の内容を記載した S O A P リクエストとを、1 つの H T T P レスポンスに記載して前記通信クライアントに送信する送信手段とを設けたことを特徴とする通信サーバ。

【請求項 9】

請求項 8 記載の通信サーバであって、

前記第 1 の記憶手段に記憶させたクライアント要求及び前記第 2 の記憶手段に記憶させたサーバ要求に優先順位を付す手段を設け、

前記応答生成手段を、前記優先順位の高いクライアント要求から順に読み出してそのクライアント要求に対する動作応答を生成し、前記第 1 の記憶手段に記憶させる手段とし、

前記収集手段を、前記優先順位の高いクライアント要求に対する動作応答から順に前記第 1 の記憶手段から読み出し、前記優先順位の高いサーバ要求から順に前記第 2 の記憶手段から読み出す手段としたことを特徴とする通信サーバ。

【請求項 1 0】

通信クライアントから通信要求を受信し、前記通信クライアントに対してその通信要求に対する通信応答を送信し、前記通信クライアントからの動作要求であるクライアント要求を記載した通信要求を前記通信クライアントから受信し、該通信クライアントに、そのクライアント要求に対する動作応答を前記通信応答に記載して送信する通信サーバの制御方法において、

前記クライアント要求と前記通信クライアントに送信した動作要求であるサーバ要求に対する動作応答とを前記通信要求に記載した状態で一括して前記通信クライアントから受信する受信手順と、

その通信要求に対する通信応答として、前記通信クライアントから受信したクライアント要求に対する動作応答と前記サーバ要求とを一括して前記通信クライアントに送信する送信手順と、

前記クライアント要求に係る動作を実行し、実行結果としてそのクライアント要求に対する動作応答を生成する手順とを前記通信サーバに実行させることを特徴とする通信サーバの制御方法。

【請求項 1 1】

請求項 1 0 記載の通信サーバの制御方法であって、

前記動作要求は関数呼び出しであり、

前記動作応答はその関数呼び出しによって呼び出された関数の実行結果であることを特徴とする通信サーバの制御方法。

【請求項 1 2】

通信クライアントから通信要求を受信し、前記通信クライアントに対してその通信要求に対する通信応答を送信し、前記通信クライアントからの動作要求であるクライアント要求を記載した通信要求を前記通信クライアントから受信し、該通信クライアントに、そのクライアント要求に対する動作応答を前記通信応答に記載して送信する通信サーバの制御方法において、

前記クライアント要求とこの要求に対する動作応答とを記憶する第 1 の記憶領域を設ける手順と、

前記通信クライアントに対する動作要求であるサーバ要求とこの要求に対する動作応答とを記憶する第 2 の記憶領域を設ける手順と、



前記サーバ要求を生成して前記第 2 の記憶領域に記憶させる要求生成手順と、

前記第 1 の記憶領域からクライアント要求を読み出し、そのクライアント要求に係る動作を実行し、その実行結果としてそのクライアント要求に対する動作応答を生成し、その動作応答を読み出したクライアント要求と関連付けて前記第 1 の記憶領域に記憶させる応答生成手順と、

前記クライアント要求と前記通信クライアントに送信したサーバ要求に対する動作応答とを前記通信要求に記載した状態で一括して前記通信クライアントから受信する受信手順と、

該受信手順で受信したクライアント要求を前記第 1 の記憶領域に記憶させると共に、前記受信手順で受信した、前記通信クライアントに送信したサーバ要求に対する動作応答を、前記通信クライアントに送信したサーバ要求と関連付けて前記第 2 の記憶領域に記憶させる分配手順と、

前記クライアント要求に対する動作応答を前記第 1 の記憶領域から読み出すと共に、前記サーバ要求を前記第 2 の記憶領域から読み出す収集手順と、

前記受信手順で受信した通信要求に対する通信応答として、前記収集手順で読み出した動作応答とサーバ要求とを一括して前記通信クライアントに送信する送信手順とを前記通信サーバに実行させることを特徴とする通信サーバの制御方法。

【請求項 1 3】

請求項 1 2 記載の通信サーバの制御方法であって、

前記受信手順において、前記通信クライアントから受信する動作応答と動作要求とを、それぞれ SOAP メッセージとして受信させるようにし、

前記送信手順において、前記通信クライアントに送信すべき動作応答と動作要求とを、それぞれ SOAP メッセージとして送信させるようにしたことを特徴とする通信サーバの制御方法。

【請求項 1 4】

請求項 1 2 又は 1 3 記載の通信サーバの制御方法であって、

前記第 1 の記憶領域に記憶させたクライアント要求及び前記第 2 の記憶領域に記憶させたサーバ要求に優先順位を付す手順を前記通信サーバに実行させ、

前記応答生成手順に、前記優先順位の高いクライアント要求から順に読み出し、そのクライアント要求に対する動作応答を生成して前記第 1 の記憶領域に記憶させるようにし、

前記収集手順に、前記優先順位の高いクライアント要求に対する動作応答から順に前記第 1 の記憶領域から読み出し、前記優先順位の高いサーバ要求から順に前記第 2 の記憶領域から読み出させるようにしたことを特徴とする通信サーバの制御方法。

【請求項 1 5】

コンピュータを、通信クライアントから通信要求を受信し、前記通信クライアントに対してその通信要求に対する通信応答を送信し、前記通信クライアントからの動作要求であるクライアント要求を記載した通信要求を前記通信クライアントから受信し、該通信クライアントに、そのクライアント要求に対する動作応答を前記通信応答に記載して送信する通信サーバとして機能させるためのプログラムにおいて、

前記コンピュータを、前記クライアント要求と前記通信クライアントに送信した動作要求であるサーバ要求に対する動作応答とを前記通信要求に記載した状態で一括して前記通信クライアントから受信する受信手段と、

その通信要求に対する通信応答として、前記通信クライアントから受信したクライアント要求に対する動作応答と前記サーバ要求とを一括して前記通信クライアントに送信する送信手段と、

前記クライアント要求に係る動作を実行し、実行結果としてそのクライアント要求に対する動作応答を生成する手段として機能させるためのプログラムを含むことを特徴とするプログラム。

【請求項 1 6】

請求項 1 5 記載のプログラムであって、

前記動作要求は関数呼び出しであり、

前記動作応答はその関数呼び出しによって呼び出された関数の実行結果であることを特徴とするプログラム。

【請求項 1 7】

コンピュータを、通信クライアントから通信要求を受信し、前記通信クライアントに対してその通信要求に対する通信応答を送信し、前記通信クライアントからの動作要求であるクライアント要求を記載した通信要求を前記通信クライアントから受信し、該通信クライアントに、そのクライアント要求に対する動作応答を前記通信応答に記載して送信する通信サーバとして機能させるためのプログラムにおいて、

前記コンピュータを、前記クライアント要求とこの要求に対する動作応答とを記憶する第 1 の記憶手段と、

前記通信クライアントに対する動作要求であるサーバ要求とこの要求に対する動作応答とを記憶する第 2 の記憶手段と、

前記サーバ要求を生成して前記第 2 の記憶手段に記憶させる要求生成手段と、

前記第 1 の記憶手段からクライアント要求を読み出し、そのクライアント要求に係る動作を実行し、その実行結果としてそのクライアント要求に対する動作応答を生成し、その動作応答を読み出したクライアント要求と関連付けて前記第 1 の記憶手段に記憶させる応答生成手段と、

前記クライアント要求と前記通信クライアントに送信したサーバ要求に対する動作応答とを前記通信要求に記載した状態で一括して前記通信クライアントから受信する受信手段と、

該受信手段が受信したクライアント要求を前記第 1 の記憶手段に記憶させると共に、前記受信手段が受信した、前記通信クライアントに送信したサーバ要求に対する動作応答を、前記通信クライアントに送信したサーバ要求と関連付けて前記第 2 の記憶手段に記憶させる分配手段と、

前記クライアント要求に対する動作応答を前記第 1 の記憶手段から読み出すと共に、前記サーバ要求を前記第 2 の記憶手段から読み出す収集手段と、

前記受信手段が受信した通信要求に対する通信応答として、前記収集手段が読み出した動作応答とサーバ要求とを一括して前記通信クライアントに送信する送信手段として機能させるためのプログラムを含むことを特徴とするプログラム。

【請求項 1 8】

請求項 1 7 記載のプログラムであって、

前記受信手段の機能を、前記通信クライアントから受信する動作応答と動作要求とを、それぞれ SOAP メッセージとして受信する機能とし、

前記送信手段の機能を、前記通信クライアントに送信すべき動作応答と動作要求とを、それぞれ SOAP メッセージとして送信する機能としたことを特徴とするプログラム。

【請求項 1 9】

請求項 1 7 又は 1 8 記載のプログラムであって、

前記第 1 の記憶手段に記憶させたクライアント要求及び前記第 2 の記憶手段に記憶させたサーバ要求に優先順位を付す手段として機能させるためのプログラムをさらに含め、

前記応答生成手段の機能を、前記優先順位の高いクライアント要求から順に読み出し、そのクライアント要求に対する動作応答を生成して前記第 1 の記憶手段に記憶させる機能とし、

前記収集手段の機能を、前記優先順位の高いクライアント要求に対する動作応答から順に前記第 1 の記憶手段から読み出し、前記優先順位の高いサーバ要求から順に前記第 2 の記憶手段から読み出す機能としたことを特徴とするプログラム。

【請求項 2 0】

請求項 1 5 乃至 1 9 のいずれか一項記載のプログラムを記録したコンピュータ読み取り可能な記録媒体。

**【書類名】 明細書****【発明の名称】 通信サーバ、通信サーバの制御方法、プログラム及び記録媒体****【技術分野】****【0 0 0 1】**

この発明は、通信クライアントから通信要求を受信し、通信クライアントに対してその通信要求に対する通信応答を送信し、通信クライアントからの動作要求を記載した通信要求を通信クライアントから受信し、通信クライアントに、その動作要求に対する動作応答を記載した通信応答を送信する通信サーバ、このような通信サーバの制御方法、コンピュータをこのような通信サーバとして機能させるためのプログラム、およびこのようなプログラムを記録したコンピュータ読み取り可能な記録媒体に関する。

**【背景技術】****【0 0 0 2】**

従来から、通信装置をネットワークを介して接続した通信システムにおいて、通信装置同士で互いにメッセージを交換させることにより、通信相手の装置に対して通知や要求を行わせることが行われている。そして、このようなシステムにおいて、ある装置から別の装置に動作要求としてコマンドを送信して動作を実行させ、送信相手から動作の実行結果を動作応答として返信させることも行われている。

また、通信システムを構成する通信装置の一部を通信クライアント、他の一部を通信サーバとし、通信クライアントと通信サーバとの間の通信を、常に通信クライアントから通信サーバに通信要求を送信し、通信サーバからその送信元の通信クライアントに対して通信応答を返すというプロトコルで行うようにすることも知られている。

そこで、通信クライアントから通信サーバへの動作要求を通信要求に記載して送信し、その動作要求に対する動作応答を通信応答に記載して通信サーバから通信クライアントに返信することも行われている。

**【0 0 0 3】**

また、逆に通信サーバから通信クライアントに動作要求を送信して動作を行わせる技術としては、以下のようなものが知られている。

例えば、特許文献 1 には、リモートプロセッサがローカルプロセッサに対して実行されるべきコマンドを指示するメッセージを送信し、そのコマンドに対する応答を受信することが記載されている。

また、この文献には、ローカルプロセッサがファイアウォールの内側に配置されている場合において、ローカルプロセッサからファイアウォールの外側のリモートプロセッサに通信要求を送信し、リモートプロセッサがこの通信要求に対する応答としてローカルプロセッサに対してコマンドを送信するようにすることにより、ファイアウォールの外側から内側に向けてコマンドを送信できるようにする技術も開示されている。

この場合において、ローカルプロセッサが通信クライアントに、リモートプロセッサが通信サーバに該当する。

**【特許文献 1】 特開 2 0 0 1 - 2 7 3 2 1 1 号公報****【0 0 0 4】**

また、このような動作要求に関する技術は、通信装置に接続された装置の動作を遠隔制御するシステムにも適用することができる。特許文献 2 には、ブラインド及び照明を操作する機能を有する遠隔被操作装置に、ユーザからの操作を受け付ける機能を有する遠隔操作装置からコマンドを送信してブラインド及び照明を操作させる遠隔操作システムにこのような技術を適用した例が記載されている。ただし、この文献には、コマンドに対する応答を送信する点は示されていない。

**【特許文献 2】 特開 2 0 0 2 - 1 3 5 8 5 8 号公報****【発明の開示】****【発明が解決しようとする課題】****【0 0 0 5】**

ところで、複数の通信装置間でメッセージを交換する場合において、コマンドを送信す



る通信装置は、1つとは限らない。複数の通信装置が互いに相手に対してコマンドを送信するようにすることも可能であり、この場合には、コマンドを受け付けた通信装置に、それぞれコマンドの送信元に対して実行結果を返させるようにすることが求められている。そして、このような動作を行う場合、ある通信装置から通信相手の装置に送信する情報としては、通信相手の装置に対するコマンドと、通信相手の装置から受信したコマンドについての実行結果とが考えられる。

#### 【0006】

従来は、これらのコマンドと実行結果とは別々に送信するようにしていた。しかし、このような方式では、コマンドの送信時と受信したコマンドに対する実行結果の送信時に、それぞれ別々に通信のコネクションを確立する必要がある。従って、通信のオーバーヘッドが大きくなり、効率性の点で問題があった。

現状では、ネットワークを介した通信をダイヤルアップ接続で行う環境もまだ多く残っており、このような環境においては上記の点が特に問題となる。このような環境では、コネクションの確立に数十秒単位の時間を要することもあり、またコネクションを確立する毎に料金を課金されるので、コネクションを確立する回数が増加するとコストアップにつながるためである。

この発明は、このような問題を解決し、通信要求とそれに対する通信応答とを送受信する複数の通信装置が互いに動作要求及び受信した動作要求に対する動作応答を送受信する場合において、通信の効率を上げることを目的とする。

#### 【課題を解決するための手段】

#### 【0007】

上記の目的を達成するため、この発明の通信サーバは、通信クライアントから通信要求を受信し、上記通信クライアントに対してその通信要求に対する通信応答を送信し、上記通信クライアントからの動作要求であるクライアント要求を記載した通信要求を上記通信クライアントから受信し、その通信クライアントに、そのクライアント要求に対する動作応答を上記通信応答に記載して送信する通信サーバにおいて、上記クライアント要求と上記通信クライアントに送信した動作要求であるサーバ要求に対する動作応答とを上記通信要求に記載した状態で一括して上記通信クライアントから受信する受信手段と、その通信要求に対する通信応答として、上記通信クライアントから受信したクライアント要求に対する動作応答と上記サーバ要求とを一括して上記通信クライアントに送信する送信手段と、上記クライアント要求に係る動作を実行し、実行結果としてそのクライアント要求に対する動作応答を生成する手段と設けたものである。

このような通信サーバにおいて、上記動作要求を関数呼び出しとし、上記動作応答をその関数呼び出しによって呼び出された関数の実行結果とするとよい。

#### 【0008】

また、この発明は、通信クライアントから通信要求を受信し、上記通信クライアントに対してその通信要求に対する通信応答を送信し、上記通信クライアントからの動作要求であるクライアント要求を記載した通信要求を上記通信クライアントから受信し、その通信クライアントに、そのクライアント要求に対する動作応答を上記通信応答に記載して送信する通信サーバにおいて、上記クライアント要求とこの要求に対する動作応答とを記憶する第1の記憶手段と、上記通信クライアントに対する動作要求であるサーバ要求とこの要求に対する動作応答とを記憶する第2の記憶手段と、上記サーバ要求を生成して上記第2の記憶手段に記憶させる要求生成手段と、上記第1の記憶手段からクライアント要求を読み出し、そのクライアント要求に係る動作を実行し、その実行結果としてそのクライアント要求に対する動作応答を生成し、その動作応答を読み出したクライアント要求と関連付けて上記第1の記憶手段に記憶させる応答生成手段と、上記クライアント要求と上記通信クライアントに送信したサーバ要求に対する動作応答とを上記通信要求に記載した状態で一括して上記通信クライアントから受信する受信手段と、その受信手段が受信したクライアント要求を上記第1の記憶手段に記憶させると共に、上記受信手段が受信した、上記通信クライアントに送信したサーバ要求に対する動作応答を、上記通信クライアントに送信

したサーバ要求と関連付けて上記第2の記憶手段に記憶させる分配手段と、上記クライアント要求に対する動作応答を上記第1の記憶手段から読み出すと共に、上記サーバ要求を上記第2の記憶手段から読み出す収集手段と、上記受信手段が受信した通信要求に対する通信応答として、上記収集手段が読み出した動作応答とサーバ要求とを一括して上記通信クライアントに送信する送信手段とを設けた通信サーバも提供する。

#### 【0009】

このような通信サーバにおいて、上記受信手段が、上記通信クライアントから受信する動作応答と動作要求とを、それぞれSOAPメッセージとして受信するようにし、上記送信手段が、上記通信クライアントに送信すべき動作応答と動作要求とを、それぞれSOAPメッセージとして送信するようにするとよい。

#### 【0010】

さらに、上記第1の記憶手段に記憶させたクライアント要求及び上記第2の記憶手段に記憶させたサーバ要求に優先順位を付す手段を設け、上記応答生成手段を、上記優先順位の高いクライアント要求から順に読み出し、そのクライアント要求に対する動作応答を生成して上記第1の記憶手段に記憶させる手段とし、上記収集手段を、上記優先順位の高いクライアント要求に対する動作応答から順に上記第1の記憶手段から読み出し、上記優先順位の高いサーバ要求から順に上記第2の記憶手段から読み出す手段とするとよい。

#### 【0011】

また、この発明は、通信クライアントからHTTPリクエストを受信し、上記通信クライアントに対してそのHTTPリクエストに対するHTTPレスポンスを送信し、上記通信クライアントからのSOAPリクエストをHTTPリクエストに記載した状態で上記通信クライアントから受信し、その通信クライアントにそのSOAPリクエストに対するSOAPレスポンスを上記HTTPレスポンスに記載して送信する通信サーバにおいて、上記通信クライアントからのSOAPリクエストと上記通信クライアントに送信したSOAPリクエストに対するSOAPレスポンスとを1つのHTTPリクエストに記載した状態で上記通信クライアントから受信する受信手段と、そのHTTPリクエストに対するHTTPレスポンスとして、上記通信クライアントからのSOAPリクエストに対するSOAPレスポンスと上記通信クライアントに対するSOAPリクエストとを、1つのHTTPレスポンスに記載して上記通信クライアントに送信する送信手段と、上記通信クライアントから受信したSOAPリクエストに係る動作を実行し、そのSOAPリクエストに対するSOAPレスポンスに記載すべき実行結果を生成する手段と設けた通信サーバも提供する。

このような通信サーバにおいて、上記SOAPリクエストには関数呼び出しを記載し、上記SOAPレスポンスにはその関数呼び出しによって呼び出された関数の実行結果を記載するようにするとよい。

#### 【0012】

さらにまた、この発明は、通信クライアントからHTTPリクエストを受信し、上記通信クライアントに対してそのHTTPリクエストに対するHTTPレスポンスを送信し、上記通信クライアントからのSOAPリクエストをHTTPリクエストに記載した状態で上記通信クライアントから受信し、その通信クライアントにそのSOAPリクエストに対するSOAPレスポンスを上記HTTPレスポンスに記載して送信する通信サーバにおいて、上記通信クライアントからの動作要求であるクライアント要求とこの要求に対する動作応答とを記憶する第1の記憶手段と、上記通信クライアントに対する動作要求であるサーバ要求とこの要求に対する動作応答とを記憶する第2の記憶手段と、上記サーバ要求を生成して上記第2の記憶手段に記憶させる要求生成手段と、上記第1の記憶手段からクライアント要求を読み出し、そのクライアント要求に係る動作を実行し、その実行結果としてそのクライアント要求に対する動作応答を生成し、その動作応答を読み出したクライアント要求と関連付けて上記第1の記憶手段に記憶させる応答生成手段と、上記クライアント要求を記載したSOAPリクエストと、上記通信クライアントに送信したSOAPリクエストに対するSOAPレスポンスであって上記通信クライアントに送信したサーバ要求

に対する動作応答を記載したSOAPレスポンスとを、1つのHTTPリクエストに記載した状態で上記通信クライアントから受信する受信手段と、その受信手段が受信したSOAPリクエストに記載されたクライアント要求の内容を上記第1の記憶手段に記憶させると共に、上記受信手段が受信したSOAPレスポンスに記載された動作応答の内容を、上記通信クライアントに送信したサーバ要求と関連付けて上記第2の記憶手段に記憶させる分配手段と、上記クライアント要求に対する動作応答を上記第1の記憶手段から読み出すと共に、上記サーバ要求を上記第2の記憶手段から読み出す収集手段と、上記1つのHTTPリクエストに対するHTTPレスポンスとして、上記収集手段が読み出した動作応答の内容を記載したSOAPレスポンスと、上記収集手段が読み出した動作要求の内容を記載したSOAPリクエストとを、1つのHTTPレスポンスに記載して上記通信クライアントに送信する送信手段とを設けた通信サーバも提供する。

#### 【0013】

このような通信サーバにおいて、上記第1の記憶手段に記憶させたクライアント要求及び上記第2の記憶手段に記憶させたサーバ要求に優先順位を付す手段を設け、上記応答生成手段を、上記優先順位の高いクライアント要求から順に読み出してそのクライアント要求に対する動作応答を生成し、上記第1の記憶手段に記憶させる手段とし、上記収集手段を、上記優先順位の高いクライアント要求に対する動作応答から順に上記第1の記憶手段から読み出し、上記優先順位の高いサーバ要求から順に上記第2の記憶手段から読み出す手段とするとよい。

#### 【0014】

また、この発明の通信サーバの制御方法は、通信クライアントから通信要求を受信し、上記通信クライアントに対してその通信要求に対する通信応答を送信し、上記通信クライアントからの動作要求であるクライアント要求を記載した通信要求を上記通信クライアントから受信し、その通信クライアントに、そのクライアント要求に対する動作応答を上記通信応答に記載して送信する通信サーバの制御方法において、上記クライアント要求と上記通信クライアントに送信した動作要求であるサーバ要求に対する動作応答とを上記通信要求に記載した状態で一括して上記通信クライアントから受信する受信手順と、その通信要求に対する通信応答として、上記通信クライアントから受信したクライアント要求に対する動作応答と上記サーバ要求とを一括して上記通信クライアントに送信する送信手順と、上記クライアント要求に係る動作を実行し、実行結果としてそのクライアント要求に対する動作応答を生成する手順とを上記通信サーバに実行させるようにしたものである。

このような通信サーバの制御方法において、上記動作要求を関数呼び出しとし、上記動作応答をその関数呼び出しによって呼び出された関数の実行結果とするとよい。

#### 【0015】

また、この発明は、通信クライアントから通信要求を受信し、上記通信クライアントに対してその通信要求に対する通信応答を送信し、上記通信クライアントからの動作要求であるクライアント要求を記載した通信要求を上記通信クライアントから受信し、その通信クライアントに、そのクライアント要求に対する動作応答を上記通信応答に記載して送信する通信サーバの制御方法において、上記クライアント要求とこの要求に対する動作応答とを記憶する第1の記憶領域を設ける手順と、上記通信クライアントに対する動作要求であるサーバ要求とこの要求に対する動作応答とを記憶する第2の記憶領域を設ける手順と、上記サーバ要求を生成して上記第2の記憶領域に記憶させる要求生成手順と、上記第1の記憶領域からクライアント要求を読み出し、そのクライアント要求に係る動作を実行し、その実行結果としてそのクライアント要求に対する動作応答を生成し、その動作応答を読み出したクライアント要求と関連付けて上記第1の記憶領域に記憶させる応答生成手順と、上記クライアント要求と上記通信クライアントに送信したサーバ要求に対する動作応答とを上記通信要求に記載した状態で一括して上記通信クライアントから受信する受信手順と、その受信手順で受信したクライアント要求を上記第1の記憶領域に記憶させると共に、上記受信手順で受信した、上記通信クライアントに送信したサーバ要求に対する動作応答を、上記通信クライアントに送信したサーバ要求と関連付けて上記第2の記憶領域に



記憶させる分配手順と、上記クライアント要求に対する動作応答を上記第 1 の記憶領域から読み出すと共に、上記サーバ要求を上記第 2 の記憶領域から読み出す収集手順と、上記受信手順で受信した通信要求に対する通信応答として、上記収集手順で読み出した動作応答とサーバ要求とを一括して上記通信クライアントに送信する送信手順とを上記通信サーバに実行させる通信サーバの制御方法も提供する。

【0 0 1 6】

このような通信サーバの制御方法において、上記受信手順において、上記通信クライアントから受信する動作応答と動作要求とを、それぞれ S O A P メッセージとして受信させるようにし、上記送信手順において、上記通信クライアントに送信すべき動作応答と動作要求とを、それぞれ S O A P メッセージとして送信させるようにするとよい。

【0 0 1 7】

さらに、上記第 1 の記憶領域に記憶させたクライアント要求及び上記第 2 の記憶領域に記憶させたサーバ要求に優先順位を付す手順を上記通信サーバに実行させ、上記応答生成手順に、上記優先順位の高いクライアント要求から順に読み出し、そのクライアント要求に対する動作応答を生成して上記第 1 の記憶領域に記憶させるようにし、上記収集手順に、上記優先順位の高いクライアント要求に対する動作応答から順に上記第 1 の記憶領域から読み出し、上記優先順位の高いサーバ要求から順に上記第 2 の記憶領域から読み出させるようにするとよい。

【0 0 1 8】

また、この発明のプログラムは、コンピュータを、通信クライアントから通信要求を受信し、上記通信クライアントに対してその通信要求に対する通信応答を送信し、上記通信クライアントからの動作要求であるクライアント要求を記載した通信要求を上記通信クライアントから受信し、その通信クライアントに、そのクライアント要求に対する動作応答を上記通信応答に記載して送信する通信サーバとして機能させるためのプログラムにおいて、上記コンピュータを、上記クライアント要求と上記通信クライアントに送信した動作要求であるサーバ要求に対する動作応答とを上記通信要求に記載した状態で一括して上記通信クライアントから受信する受信手段と、その通信要求に対する通信応答として、上記通信クライアントから受信したクライアント要求に対する動作応答と上記サーバ要求とを一括して上記通信クライアントに送信する送信手段と、上記クライアント要求に係る動作を実行し、実行結果としてそのクライアント要求に対する動作応答を生成する手段として機能させるためのプログラムを含めたものである。

このようなプログラムにおいて、上記動作要求を関数呼び出しとし、上記動作応答をその関数呼び出しによって呼び出された関数の実行結果とするとよい。

【0 0 1 9】

さらに、この発明は、コンピュータを、通信クライアントから通信要求を受信し、上記通信クライアントに対してその通信要求に対する通信応答を送信し、上記通信クライアントからの動作要求であるクライアント要求を記載した通信要求を上記通信クライアントから受信し、その通信クライアントに、そのクライアント要求に対する動作応答を上記通信応答に記載して送信する通信サーバとして機能させるためのプログラムにおいて、上記コンピュータを、上記クライアント要求とこの要求に対する動作応答とを記憶する第 1 の記憶手段と、上記通信クライアントに対する動作要求であるサーバ要求とこの要求に対する動作応答とを記憶する第 2 の記憶手段と、上記サーバ要求を生成して上記第 2 の記憶手段に記憶させる要求生成手段と、上記第 1 の記憶手段からクライアント要求を読み出し、そのクライアント要求に係る動作を実行し、その実行結果としてそのクライアント要求に対する動作応答を生成し、その動作応答を読み出したクライアント要求と関連付けて上記第 1 の記憶手段に記憶させる応答生成手段と、上記クライアント要求と上記通信クライアントに送信したサーバ要求に対する動作応答とを上記通信要求に記載した状態で一括して上記通信クライアントから受信する受信手段と、その受信手段が受信したクライアント要求を上記第 1 の記憶手段に記憶させると共に、上記受信手段が受信した、上記通信クライアントに送信したサーバ要求に対する動作応答を、上記通信クライアントに送信したサーバ



要求と関連付けて上記第 2 の記憶手段に記憶させる分配手段と、上記クライアント要求に対する動作応答を上記第 1 の記憶手段から読み出すと共に、上記サーバ要求を上記第 2 の記憶手段から読み出す収集手段と、上記受信手段が受信した通信要求に対する通信応答として、上記収集手段が読み出した動作応答とサーバ要求とを一括して上記通信クライアントに送信する送信手段として機能させるためのプログラムを含めたプログラムも提供する。

#### 【 0 0 2 0 】

このようなプログラムにおいて、上記受信手段の機能を、上記通信クライアントから受信する動作応答と動作要求とを、それぞれ S O A P メッセージとして受信する機能とし、上記送信手段の機能を、上記通信クライアントに送信すべき動作応答と動作要求とを、それぞれ S O A P メッセージとして送信する機能とするとよい。

#### 【 0 0 2 1 】

さらに、上記第 1 の記憶手段に記憶させたクライアント要求及び上記第 2 の記憶手段に記憶させたサーバ要求に優先順位を付す手段として機能させるためのプログラムを含め、上記応答生成手段の機能を、上記優先順位の高いクライアント要求から順に読み出し、そのクライアント要求に対する動作応答を生成して上記第 1 の記憶手段に記憶させる機能とし、上記収集手段の機能を、上記優先順位の高いクライアント要求に対する動作応答から順に上記第 1 の記憶手段から読み出し、上記優先順位の高いサーバ要求から順に上記第 2 の記憶手段から読み出す機能とするとよい。

#### 【 0 0 2 2 】

また、この発明の記録媒体は、これらの何れかのプログラムを記録したコンピュータ読み取り可能な記録媒体である。

#### 【発明の効果】

#### 【 0 0 2 3 】

以上のようなこの発明の通信サーバおよび通信サーバの制御方法によれば、通信要求とそれに対する通信応答とを送受信する複数の通信装置が互いに動作要求及び受信した動作要求に対する動作応答を送受信する通信システムを構成する場合において、通信の効率を上げることができる。また、この発明のプログラムによれば、コンピュータを上記の通信装置として機能させてその特徴を実現し、同様な効果を得ることができる。この発明の記録媒体によれば、上記のプログラムを記憶していないコンピュータにそのプログラムを読み出させて実行させ、上記の効果を得ることができる。

#### 【発明を実施するための最良の形態】

#### 【 0 0 2 4 】

以下、この発明を実施するための最良の形態について、図面を参照して説明する。

まず図 1 に、この発明の通信装置を用いて構成したこの発明の通信システムの構成例を示す。

この通信システムは、図 1 に示すように、それぞれこの発明の通信装置である第 1 の通信装置 1 と第 2 の通信装置 2 とをネットワーク 1 0 によって接続して構成している。

そして、第 1 の通信装置 1 及び第 2 の通信装置 2 は、通信機能を備えた P C 等のコンピュータを始め、通信機能及び情報処理機能を備えた各種電子装置として構成することができる。ネットワーク 1 0 としては、インターネットや L A N (ローカルエリアネットワーク) を始め、有線、無線を問わず、ネットワーク通信が可能な各種通信経路を用いることができる。

#### 【 0 0 2 5 】

また、第 1 の通信装置 1 及び第 2 の通信装置 2 は、互いの制御管理を行うためのアプリケーションプログラムを実装している。そして、これらの各ノードは、R P C (Remote Procedure Call) により、互いの実装するアプリケーションプログラムのメソッドに対する処理の依頼である「動作要求」を送信し、この依頼された処理の結果である「動作応答」を取得することができるようになっている。即ち、第 1 の通信装置 1 は、第 2 の通信装置 2 への要求 (以下、第 1 の通信装置側要求という) を生成してこれを第 2 の通信装置 2

へ引き渡し、この要求に対する応答を取得できる一方で、第 2 の通信装置 2 は、第 1 の通信装置 1 への要求（以下、第 2 の通信装置側要求という）を生成してこれを第 1 の通信装置 1 へ引き渡し、この要求に対する応答を取得できるようになっている。

なお、ここではメソッドを入力と出力の形式を規定した論理的な関数として定義するものとする。そしてこの場合、動作要求はこの関数を呼び出す関数呼び出し（Procedure Call）となり、動作応答はその関数呼び出しによって呼び出された関数の実行結果となる。

#### 【 0 0 2 6 】

図 2 に、これらの動作要求と動作応答の関係を示す。

図 2（A）は、第 1 の通信装置 1 で第 2 の通信装置 2 に対する動作要求が発生したケースである。このケースでは、第 1 の通信装置 1 が第 1 の通信装置側動作要求を生成して第 2 の通信装置 2 に送信し、これを受け取った第 2 の通信装置 2 がその要求に対する動作応答を返すというモデルになる。

#### 【 0 0 2 7 】

図 2（B）は、第 2 の通信装置 2 で第 1 の通信装置 1 に対する動作要求が発生したケースである。このケースでは、第 2 の通信装置 2 が第 2 の通信装置側要求を生成して第 1 の通信装置 1 に送信し、これを受け取った第 1 の通信装置 1 がその要求に対する動作応答を返すというモデルになる。

なお、ここでは R P C による引数並びに戻り値の受け渡しのプロトコルとして S O A P（Simple Object Access Protocol）を採用し、上記の動作要求や動作応答は、ここでは S O A P メッセージとして記載するようにしている。

#### 【 0 0 2 8 】

この発明の特徴は、このように複数の通信装置が互いに動作要求及び受信した動作要求に対する動作応答を送受信する場合において、通信相手の装置に送信すべき動作要求とその通信相手の装置から受信した動作要求に対する動作応答とを一括して送信するようにする点である。

そして、実際に動作要求や動作応答を転送するための通信プロトコルとしては、システムの構成に合わせて適当なものを採用することができ、例えば H T T P（HyperText Transfer Protocol）や S M T P（Simple Mail Transfer Protocol）を採用することができる。ただし、S M T P によって転送される電子メールには通信要求と通信応答のような対応関係がないため、S M T P を採用した装置は、この発明の通信サーバの実施例には含まれない。

そこで、まず通信プロトコルに H T T P を採用する場合の実施例について説明し、次に S M T P を採用する場合についても参考例として説明する。

#### 【 0 0 2 9 】

〔H T T P を採用する場合の実施例：図 3 乃至図 2 1〕

図 3 に、H T T P を採用する場合の実施例を適用する通信システムの構成例を示す。

この通信システムは、図 3 に示すように、H T T P サーバ 1 2 と H T T P クライアント 1 1 とをインターネット 1 3 によって接続して構成している。ただし、セキュリティを向上させるため、H T T P クライアント 1 1 はファイアウォール 1 4 を介してインターネット 1 3 に接続するようにしている。そして、H T T P サーバ 1 2 が通信サーバであって第 1 の通信装置に該当し、H T T P クライアント 1 1 が通信クライアントであって第 2 の通信装置に該当する。

#### 【 0 0 3 0 】

なお、H T T P を用いて通信を行う場合、ファイアウォール 1 4 の内側にあるノードに対しては、ファイアウォール 1 4 の外側からは自由にアクセスできず、そのノードからの通信要求（H T T P リクエスト）に対する通信応答（H T T P レスポンス）という形でしかデータを送信できない。そこで、この通信システムにおいては、ファイアウォール 1 4 の内側にあるノードを H T T P クライアント 1 1、外側にあるノードを H T T P サーバ 1 2 としているのである。従って、これらの各ノードの機能は、これら相互間の通信以外においては、クライアントあるいはサーバに限定する必要はない。

**【0031】**

また、HTTPサーバ12及びHTTPクライアント11は、図1に示した第1の通信装置1及び第2の通信装置2の場合と同様に、互いの制御管理を行うためのアプリケーションプログラムを実装している。そして、RPC (Remote Procedure Call) により、互いの実装するアプリケーションプログラムのメソッドに対する処理の依頼である「動作要求」を送信し、この依頼された処理の結果である「動作応答」を取得することができるようになっている。

**【0032】**

図4に、これらの動作要求と動作応答の関係を示す。

図4 (A) は、HTTPクライアント11でHTTPサーバ12に対する動作要求が発生したケースである。このケースでは、HTTPクライアント11がクライアント側動作要求（クライアント要求に該当する。以下、「クライアントコマンド」とも呼ぶ）を生成してHTTPサーバ12に送信し、これを受け取ったHTTPサーバ12がそのコマンドに対する動作応答（以下、「コマンド応答」あるいは単に「応答」とも呼ぶ）を返すというモデルになる。

**【0033】**

図4 (B) は、HTTPサーバ12でHTTPクライアント11に対する動作要求が発生したケースである。このケースでは、HTTPサーバ12がサーバ側動作要求（サーバ要求に該当する。以下、「サーバコマンド」とも呼ぶ）を生成してHTTPクライアント11に送信し、これを受け取ったHTTPクライアント11がそのコマンドに対する動作応答を返すというモデルになる。

このように、動作要求及び動作応答は、RPCのレベルではHTTPクライアント11とHTTPサーバ12との間で対称に取り扱われるものである。しかし、通信のレベルでは対称ではない。

**【0034】**

図5にこの通信システムにおける通信シーケンスの例を示す。

この図に示すように、この通信システムにおいては、通信は常に、HTTPクライアント11から通信要求としてHTTPリクエストをHTTPサーバ12に送信し、HTTPサーバ12からこの通信要求に対する通信応答としてHTTPレスポンスをHTTPクライアント11に返すという手順で行われる。例えばHTTPクライアント11が送信したHTTPリクエストXに対してHTTPサーバ12がHTTPレスポンスXを返し、同じくHTTPリクエストYに対してHTTPレスポンスYを返すという具合である。

**【0035】**

そして、HTTPリクエストには、HTTPクライアント11からHTTPサーバ12に送信する動作要求であるクライアントコマンドと、HTTPサーバ12からHTTPクライアント11に送信されてきたサーバコマンドに対する応答（コマンド応答）とを記載して送信するようにしている。また、HTTPレスポンスには、HTTPサーバ12からHTTPクライアント11に送信する動作要求であるサーバコマンドと、HTTPクライアント11からHTTPサーバ12に送信されてきたクライアントコマンドに対する応答（コマンド応答）とを記載して送信するようにしている。

**【0036】**

従って、例えばクライアントコマンドAは、HTTPリクエストXに記載して転送し、コマンド応答をそのHTTPリクエストXと対応するHTTPレスポンスXに記載して転送することができる。しかし、サーバコマンドCについては、HTTPリクエストXと対応するHTTPレスポンスXに記載して転送し、そのコマンド応答は次のHTTPリクエストであるHTTPリクエストYに記載して転送することになる。

**【0037】**

また、上記図4 (A) のケースでは、クライアントコマンドが生成された後直ちにHTTPクライアント11がHTTPサーバ12とコネクションを確立し、HTTPリクエストにこれを含めて引き渡すことができるが、上記図4 (B) のケースでは、HTTPクラ



クライアント 11 側に設置されたファイアウォール 14 が HTTP サーバ 12 からの HTTP リクエストを遮断するため、HTTP サーバ 12 側から HTTP クライアント 11 へアクセスしてサーバコマンドを直ちに引き渡すことができない。

#### 【0038】

なお、クライアントコマンド及びサーバコマンドに対する応答をそれぞれ任意の数ずつ（0 でもよい）1 つの HTTP リクエストに記載することができ、サーバコマンド及びクライアントコマンドに対する応答をそれぞれ任意の数ずつ（0 でもよい）1 つの HTTP レスポンスに記載することができる。そして、1 つの HTTP リクエスト又は HTTP レスポンスに記載した内容は、論理的に一括して転送する。

そして、このようにすることにより、必要な情報を転送するために必要な接続の回数を減らし、オーバーヘッドを低減して通信の効率化を図っている。

#### 【0039】

図 6 にこの通信システムにおける別の通信シーケンスの例を示す。

説明のため、図 5 には極めて単純なシーケンス例を示したが、図 6 には、各 HTTP リクエストや HTTP レスポンスに記載するコマンドやコマンド応答の数が一定でない例を示している。

また、コマンドを受信した場合に、次の送信機会の時点で応答を返す必要もない。例えば、図 6 に示すクライアントコマンド B のように、コマンドを記載した HTTP リクエスト X' に対応する HTTP レスポンス X' に記載して応答を返さず、後の HTTP レスポンス Y' に記載して応答を返すようにしてもよい。

もちろんサーバコマンドについても同様であり、サーバコマンドを記載した HTTP レスポンスの次の HTTP リクエストにそのコマンドに対する応答を記載する必要はない。そして、さらに後の HTTP リクエストに記載して転送すればよい。

#### 【0040】

ところで、各コマンド及びコマンド応答は、それぞれ独立して生成され、また処理に供されるべきものであるから、上記のような一括転送を行うためには、転送前にこれらのコマンドやコマンド応答を結合し、また転送後に分離する処理が必要となる。次に、HTTP クライアント 11 及び HTTP サーバ 12 のハードウェア構成と共に、このような処理を行うための機能構成及びその処理の手順について説明する。

#### 【0041】

図 7 は、HTTP クライアント 11 及び HTTP サーバ 12 のハードウェア構成例を示す図である。

この図に示すように、HTTP クライアント 11 及び HTTP サーバ 12 においてはそれぞれ、CPU 31、ROM 32、RAM 33、SD (Secure Digital) カード 34、ネットワークインタフェースカード (NIC) 35 を備え、これらがシステムバス 36 で接続されている。

#### 【0042】

これら構成要素を更に具体的に説明すると、まず CPU 31 は、ROM 32 に格納している制御プログラムによって HTTP クライアント 11 又は HTTP サーバ 12 全体を統括的に制御する制御手段である。そして、ROM 32 は、CPU 31 が使用する制御プログラムを含む各種固定データを格納している読み出し専用メモリである。

RAM 33 は、CPU 31 がデータ処理を行う際のワークメモリ等として使用する一時記憶用メモリである。SD カード 34 は、装置の電源がオフになっても記憶内容を保持するようになっている不揮発性メモリである。NIC 35 は、インターネット 13 を始めとするネットワークを介して通信相手と情報の送受信を行うための通信手段である。

#### 【0043】

図 8 は、HTTP クライアント 11 の機能のうち、コマンド及びコマンド応答に関する処理を行うための機能の構成を示す機能ブロック図である。

図 8 に示す機能のうち、クライアントコマンドプール 41 及びサーバコマンドプール 42 は、いずれかの書き換え可能な記憶手段に設けられるものである。例えば SD カード 3



4 に設けることができるが、RAM 33 や図示しない HDD (ハードディスクドライブ) に設けてもよい。クライアントコマンド生成手段 43、サーバコマンド実行結果生成手段 44、送信メッセージ収集手段 45、受信メッセージ分配手段 48 の機能は、CPU 31 によって実現されるものである。また、HTTP リクエスト送信手段 46 及び HTTP レスポンス受信手段 47 の機能は、CPU 31 及び NIC 35 によって実現されるものである。

#### 【0044】

これらの機能についてさらに詳述する。

まず、クライアントコマンドプール 41 は、HTTP クライアント 11 に設けた第 2 の記憶領域に該当し、クライアントコマンドと、このコマンドに対する応答と、このコマンドの識別情報とを関連付けて登録するプールである。また、サーバコマンドプール 42 は、HTTP クライアント 11 に設けた第 1 の記憶領域に該当し、サーバコマンドと、このコマンドに対する応答と、このコマンドの識別情報とを関連付けて登録するプールである。これらのプールにおいては、コマンド毎にテーブル形式のコマンドシートを作成して情報を格納することにより、コマンドと、識別情報や応答等の情報とを関連付けるようにしている。また、これらのプールを設けた記憶手段がそれぞれ HTTP クライアント 11 の第 2、第 1 の記憶手段に該当するものとする。

#### 【0045】

クライアントコマンド生成手段 43 は、要求生成手段に該当する。そして、クライアントコマンドを生成し、このコマンドを識別する識別情報 (ID) を割り当て、さらにこのコマンドを管理するための管理情報を付し、これらの情報を関連付けてテーブル形式のクライアントコマンドシートとしてクライアントコマンドプール 41 に登録する機能を有する。このうち、クライアントコマンドを生成する部分には、例えば HTTP クライアント 11 に備えるアプリケーションが該当する。また、クライアントコマンド生成手段 43 に、HTTP サーバ 12 に各コマンドを実行させる際の優先順位を、生成したクライアントコマンドに付する機能を設けてもよい。

#### 【0046】

ここで、図 9 に HTTP クライアント 11 のクライアントコマンドシートにおけるデータ構造の例を示す。

この図に示すように、HTTP クライアント 11 においては、クライアントコマンドシートには、「コマンド ID」、「メソッド名」、「入力パラメータ」、「状態」、「クライアントコマンド実行結果の通知先」、および「出力パラメータ」のデータを記憶する領域を設けている。そして、このうち「コマンド ID」、「メソッド名」、および「入力パラメータ」がクライアントコマンド (及びそこに付された ID) に該当し、「状態」及び「クライアントコマンド実行結果の通知先」が管理情報に該当する。「出力パラメータ」は、HTTP サーバ 12 から受信するコマンド応答の内容である。

#### 【0047】

次に、各項目の内容について説明する。

まず、「メソッド名」は、HTTP サーバ 12 に対するリクエストの内容であり、HTTP サーバ 12 において呼び出す関数の種類を示す。「入力パラメータ」は、「メソッド名」に付随するデータであり、関数を呼び出す際の引数である。「コマンド ID」は、クライアントコマンドを識別するための識別情報である。「状態」は、クライアントコマンドに関する処理の進行状況を示すデータであり、処理の進行と共に、「未送信」→「応答待ち」→「応答受信済」と遷移していく。

#### 【0048】

「クライアントコマンド実行結果の通知先」は、そのシートに記載しているクライアントコマンドに対する応答を受信した場合に、その旨を通知して必要な処理を実行させるモジュールを示す参照情報である。参照するモジュールは、クライアントコマンドを生成したアプリケーションであることが多いが、必ずしもそうである必要はない。「出力パラメータ」には、コマンド応答を受け取った段階で、その内容を格納する。HTTP サーバ 1

2 からのコマンド応答を受け取るまでは空である。

#### 【0049】

図8の説明に戻ると、サーバコマンド実行結果生成手段44は、応答生成手段に該当する。そして、サーバコマンドプール42からサーバコマンドを読み出して実行するアプリケーションである。そして、サーバコマンドに対する応答を生成し、サーバコマンドのコマンドIDと関連付けてサーバコマンドプール42に登録する機能を有する。なお、HTTPサーバ12から受信したサーバコマンドは、このコマンドを識別するID及びこのコマンドを管理するための管理情報と関連付けて、テーブル形式のサーバコマンドシートとしてサーバコマンドプール42に登録しておくようにしている。そして、サーバコマンド実行結果生成手段44が生成したコマンド応答も、実行したサーバコマンドについてのサーバコマンドシートに登録する。

#### 【0050】

また、サーバコマンド実行結果生成手段44に、サーバコマンドプール42から複数の種類のサーバコマンドを読み出し、各サーバコマンドに対する応答を生成する機能を設けることが考えられる。さらに、サーバコマンドがHTTPクライアント11に優先して処理を実行させるための実行優先順位の情報を含む場合には、優先順位の高いものから優先的に読み出して実行する機能を設けることも考えられる。

なお、サーバコマンド実行結果生成手段44は、アプリケーションそのものではなく、サーバコマンドの実行に必要なアプリケーションを呼び出してコマンドを実行させるモジュールであってもよい。

#### 【0051】

ここで、図10にHTTPクライアント11のサーバコマンドシートにおけるデータ構造の例を示す。

この図に示すように、HTTPクライアント11においては、サーバコマンドシートには、「コマンドID」、「メソッド名」、「入力パラメータ」、「状態」、「出力パラメータ」、および「サーバコマンドの通知先」のデータを記憶する領域を設けている。そして、このうち「コマンドID」、「メソッド名」、および「入力パラメータ」がサーバコマンド（及びそこに付されたID）に該当し、「状態」及び「サーバコマンドの通知先」が管理情報に該当する。「出力パラメータ」は、サーバコマンドの実行結果であり、HTTPクライアント11が返すコマンド応答の内容となる。

#### 【0052】

次に、各データの内容について説明する。

まず、「メソッド名」は、HTTPクライアント11に対するリクエストの内容であり、HTTPクライアント11において呼び出す関数の種類を示す。「入力パラメータ」は、「メソッド名」に付随するデータであり、関数を呼び出す際の引数である。「コマンドID」は、サーバコマンドを識別するための識別情報である。「状態」は、サーバコマンドに関する処理の状態を示すデータであり、処理の進行と共に、「未処理」→「処理完了」→「応答済」、あるいは「未処理」→「処理中」→「処理完了」→「応答済」と遷移していく。「出力パラメータ」には、サーバコマンド実行結果生成手段44によって生成された応答が格納される。サーバコマンドの実行が終了し、上記の「状態」が「処理完了」となるまでは空である。「サーバコマンドの通知先」は、サーバコマンドの実行を行うモジュールを示す参照情報である。

#### 【0053】

再び図8の説明に戻ると、送信メッセージ収集手段45は、収集手段に該当する。そして、サーバコマンド実行結果生成手段44が生成したコマンド応答とこのコマンド応答に対応するサーバコマンドのコマンドIDとを関連付けてサーバコマンドプール42から読み出すと共に、クライアントコマンド生成手段43が生成したクライアントコマンドとこのコマンドのコマンドIDとを関連付けてクライアントコマンドプール41から読み出し、これらから送信メッセージを生成する機能を有する。

なお、コマンド応答やクライアントコマンドに実行優先順位が指定されている場合には

、送信メッセージ収集手段 4 5 がそれぞれ実行優先順位の高いものから順に読み出すようにすることが考えられる。

#### 【0 0 5 4】

ここで、送信メッセージとは、上記のコマンド応答やコマンドとコマンド ID とを、構造化言語である XML (Extensible Markup Language) で、SOAP メッセージとして記載したものである。そして、送信メッセージ収集手段 4 5 は、1 つのコマンド応答あるいはコマンドにつき、送信メッセージとして 1 つの SOAP メッセージを生成する。またこのとき、各コマンドのコマンド ID は SOAP ヘッダに記載し、コマンド応答及びクライアントコマンドの内容は SOAP ボディに記載する。SOAP による通信では、SOAP ヘッダと SOAP ボディとからなる SOAP エンベロープ (封筒) と呼ばれるメッセージを XML で記載し、HTTP などのプロトコルで交換することになる。

このようなコマンドやコマンド応答からの SOAP メッセージの生成は、WSDL (Web Service Description Language) に基づいて生成される所要の変換プログラム (シリアライザ) を実行し、データを直列化することによって行うことができる。

#### 【0 0 5 5】

そして、HTTP リクエスト送信手段 4 6 は、送信手段に該当し、送信メッセージ収集手段 4 5 が生成した送信メッセージを含む HTTP リクエストを生成し、HTTP サーバ 1 2 に送信する機能を有する。このとき、1 つの HTTP リクエストに送信メッセージをいくつ含めてもよいし、コマンド応答に係る送信メッセージとクライアントコマンドに係る送信メッセージとを任意に混在させることもできる。

そこで、HTTP リクエスト送信手段 4 6 は、これらのいずれに係る送信メッセージに関わり無く、送信メッセージ収集手段 4 5 が生成した全ての送信メッセージを 1 つの HTTP リクエストに含めて送信するようにしている。ただし、1 つの HTTP リクエストに含める送信メッセージの数に上限を設けることも考えられる。

#### 【0 0 5 6】

ところで、この HTTP リクエストの送信は、送信メッセージ収集手段 4 5 がクライアントコマンドやコマンド応答等の読み出しを試みた場合には、読み出すデータがなく、結果的に送信すべき SOAP メッセージを生成しなかった場合にも行うものである。そして、この読み出しの試みは、定期的に行うものとする。例えば、タイマによって 6 0 分毎に読み出すことが考えられる。

このようにするのは、上述のように、HTTP サーバ 1 2 から HTTP クライアント 1 1 に送信したい情報があっても HTTP クライアント 1 1 から通信を要求しない限り送信できないためである。HTTP クライアント 1 1 から何も送信するデータがなかったとしても、定期的に HTTP サーバ 1 2 に対して通信要求を送信して、HTTP サーバ 1 2 から HTTP クライアント 1 1 に情報を送信する機会を与えることにより、転送の必要な情報が長期間に亘って HTTP サーバ 1 2 に滞留してしまうことを防止できる。

#### 【0 0 5 7】

なお、送信メッセージ収集手段 4 5 による読み出しと、それに続く HTTP リクエスト送信手段 4 6 による HTTP リクエストの送信とを、定期的なタイミング以外に適宜行ってよいことはもちろんである。例えば、緊急に送信が必要な情報がいずれかのプールに登録された場合に、クライアントコマンド生成手段 4 3 あるいはサーバコマンド実行結果生成手段 4 4 が送信メッセージ収集手段 4 5 にその旨を通知して読み出しを行わせるようにしてもよい。

#### 【0 0 5 8】

次に、HTTP レスポンス受信手段 4 7 は、受信手段に該当し、HTTP サーバ 1 2 から HTTP レスポンスを受信する機能を有する。そしてここでは、HTTP レスポンスには、サーバコマンド及びそのコマンドと関連付けられたコマンド ID を含む受信メッセージと、クライアントコマンドに対する応答及びそのコマンドと関連付けられたコマンド ID を含む受信メッセージとが、任意に混在して含まれている。

ここで、受信メッセージとは、上記のコマンドや応答とコマンド ID とを SOAP メッ



ページとして記載したものである。

#### 【0059】

受信メッセージ分配手段48は、分配手段に該当する。そして、HTTPレスポンス受信手段47が受信したHTTPレスポンスに含まれるデータを、クライアントコマンドプール41及びサーバコマンドプール42に振り分けて登録する機能を有する。

具体的には、サーバコマンド及びそのコマンドと関連付けられたコマンドIDとをサーバコマンドプール42にサーバコマンドシートを設けて登録すると共に、クライアントコマンドに対する応答については、そのコマンドと関連付けられたコマンドIDをクライアントコマンドプール41に記憶しているクライアントコマンドシートのコマンドIDと照合して対応するクライアントコマンドを特定し、そのクライアントコマンドについての「出力パラメータ」として登録する。

そしてこのとき、HTTPレスポンスを分割してそこに含まれる各受信メッセージを取り出し、そのデータをテーブルへの登録に必要な形式に変換するが、この変換は、WSDLに基づいて生成される所要の変換プログラム（デシリアライザ）を実行することによって行うことができる。

#### 【0060】

次に、このような機能を有するHTTPクライアント11がHTTPサーバ12に送信するHTTPリクエストの例を図11に示す。

このHTTPリクエストは、図11に示すように、ボディ部としてMIME (Multipurpose Internet Mail Extension) に従ったマルチパートのメッセージが記載され、この各パートには、それぞれエンティティヘッダが記載されると共に、詳細な図示は省略しているが、SOAPエンベロープが埋め込まれている。図11の例では、HTTPリクエストのHTTPボディには、「MIME\_boundary」で区分された各要素が、独立した第1パート、第2パート、第3パート、第4パートを構成しているが、HTTPボディに含めることのできるパート数は4つに限られない。0個を含め、いくつでもよい。

HTTPリクエストに埋め込まれて引き渡されるSOAPエンベロープには、クライアントコマンドを記載したものと、サーバコマンドに対する応答を記載したものとがある。

#### 【0061】

また、このような機能を有するHTTPクライアント11がHTTPサーバ12から受信するHTTPレスポンスの例を図12に示す。

図12に示すように、このHTTPレスポンスは、形式としては図11に示したHTTPリクエストとHTTPヘッダ部が異なるのみであり、ボディ部にはHTTPリクエストの場合と同様に、詳細な図示は省略しているが、MIMEに従ったマルチパートのSOAPエンベロープが記載される。SOAPエンベロープの内容については、当然コマンドやコマンド応答の内容に従って異なるものである。

HTTPレスポンスに埋め込まれて引き渡されるSOAPエンベロープには、サーバコマンドを記載したものと、クライアントコマンドに対する応答を記載したものとがある。

#### 【0062】

次に、これらのHTTPリクエスト又はHTTPレスポンスに記載されるパートの具体例を図13乃至図16に示す。

図13に示すのは、クライアントコマンドを記載したパートの例である。

この例においては、まず、エンティティヘッダの部分の「X-SOAP-Type」ヘッダに、このパートに記載されているSOAPメッセージがSOAPリクエストであるかSOAPレスポンスであるかを示す情報を記載している。この例では、値の「Request」により、SOAPリクエストであること、すなわちコマンドを記載したSOAPメッセージであることを示している。

また、「SOAPAction」ヘッダは、SOAPリクエストの内容を示すものであり、この例では、「http://www.…」というURI (Uniform Resource Identifier) によりリクエストの内容を示している。なお、「SOAPAction」ヘッダは、SOAPメッセージがSOAPレスポンスである場合には付加しないため、メッセージの受信側において、このヘッダの



有無により、SOAPメッセージがSOAPリクエストであるかSOAPレスポンスであるかを判断することもできる。

#### 【0063】

そして、「Envelope」タグの属性として、名前空間の宣言を行っている。そしてここでは、SOAPで標準として定義されている名前空間の他に、「http://www.foo.com/header」及び「http://www.foo.com/server」のURIで特定される名前空間の宣言を行っている。従って、「n」の名前空間接頭辞が付されたXMLタグについては「http://www.foo.com/header」のURIで特定される名前空間に属するタグであることがわかり、「ns」の名前空間接頭辞が付されたXMLタグについては「http://www.foo.com/server」のURIで特定される名前空間に属するタグであることがわかる。

#### 【0064】

またSOAPヘッダには、「要求ID」のXMLタグの内容として、このクライアントコマンドのIDである「12345」が記載されている。そして、SOAPボディには、クライアントコマンドシートの「メソッド名」に記憶されていたメソッドを指定する情報として、「異常通知」タグが記載され、その下位のタグ「エラーID」や「説明」の要素として、「入力パラメータ」に記憶されていた引数が記載されている。ここでは異常通知の通知内容が記載されている。

#### 【0065】

図14に示すのは、クライアントコマンドに対する応答を記載したパートの例である。この例においては、まず、エンティティヘッダの部分の「X-SOAP-Type」ヘッダの値を「Response」と記載することにより、このパートに記載されているSOAPメッセージがSOAPレスポンスであること、すなわちコマンド応答を記載したSOAPメッセージであることを示している。

また、この例においても、名前空間の宣言は図13に示した例と同様である。そして、SOAPヘッダには、「コマンドID」のXMLタグの内容として、応答を生成したクライアントコマンドのIDである「12345」が記述されている。SOAPボディには、「異常通知」コマンドに対する応答であることを示すための「異常通知Response」タグが設けられ、その下位のタグに、コマンド応答の内容が記載される。ここでは、異常通知を正常に受信した旨の情報が記載されている。そして、この情報がクライアントコマンドシートの「出力パラメータ」の項目に格納される。

#### 【0066】

図15に示すのは、サーバコマンドを記載したパートの例である。

この例においても、図13の場合と同様に、「X-SOAP-Type」ヘッダの値の「Request」により、このパートに記載されているSOAPエンベロープがSOAPリクエストであることを示し、「SOAPAction」ヘッダの情報により、SOAPリクエストの内容を示している。

#### 【0067】

また、「Envelope」タグの属性として、名前空間の宣言を行っている点も、図13の場合と同様である。そしてここでは、SOAPで標準として定義されている名前空間の他に、「http://www.foo.com/header」及び「http://www.foo.com/client」のURIで特定される名前空間の宣言を行っている。

SOAPヘッダには、「要求ID」のXMLタグの内容として、このクライアントコマンドのIDである「98765」が記載されている。そして、SOAPボディには、サーバコマンドシートの「メソッド名」に記憶されるべきメソッドを指定する情報として、「温度センサ値取得」タグが記載され、その下位のタグ「センサID」の要素として、「入力パラメータ」に記憶されるべき引数が記載されている。ここではセンサ値を取得するセンサのIDが記載されている。

なお、サーバがこのようなコマンドを送信する場合としては、例えば、クライアントからの異常通知を受けて異常の原因を特定しようとする場合等が考えられる。

#### 【0068】

図16に示すのは、サーバコマンドに対する応答を記載したパートの例である。

この例においても、図14の場合と同様に、エンティティヘッダの部分の「X-Soap-Type」ヘッダの値を「Response」と記載することにより、このパートに記載されているSOAPメッセージがSOAPレスポンスであることを示している。

また、この例においても、名前空間の宣言は図15に示した例と同様である。そして、SOAPヘッダには、「コマンドID」のXMLタグの内容として、応答を生成したサーバコマンドのIDである「98765」が記述されている。SOAPボディには、「温度センサ値取得」コマンドに対する応答であることを示すための「温度センサ値取得Response」タグが設けられ、その下位のタグに、コマンド応答の内容が記載される。ここでは、値取得を要求されたセンサの示す温度値の情報が記載されている。

#### 【0069】

次に、以上説明したような構成及び機能を有するHTTPクライアント11において実行する処理について、図17乃至図21のフローチャートを用いて説明する。これらのフローチャートに示す処理は、HTTPクライアント11のCPU31が所要の制御プログラムを実行することによって行うものである。

#### 【0070】

まず、図17にメッセージの収集及び分配処理の基本動作のフローチャートを示す。

HTTPクライアント11のCPU31は、送信メッセージ収集手段45がクライアントコマンドやコマンド応答等の読み出しを試みるタイミングになると、図17のフローチャートに示す処理を開始する。

そして、まずクライアントコマンドの収集処理を行う(S11)。この処理は、クライアントコマンドプール41からHTTPサーバ12に送信すべきクライアントコマンドを収集する処理であり、収集したデータからSOAPエンベロープによるパートを生成する処理を含む。

#### 【0071】

次に、サーバコマンドに対する応答であるサーバコマンド実行結果の収集処理を行う(S12)。この処理は、サーバコマンドプールからHTTPサーバ12に送信すべきコマンド応答を収集する処理であり、やはり収集したデータからSOAPエンベロープによるパートを生成する処理を含む。

その後、ステップS11及びS12の処理で生成したパートを1つにマージして、すべてのパートを含むHTTPリクエストを生成し(S13)、そのHTTPリクエストをHTTPサーバ12に送信する(S14)。

ここまでの処理において、ステップS11及びS12ではCPU31は送信メッセージ収集手段45として機能し、ステップS13及びS14ではHTTPリクエスト送信手段46として機能する。

#### 【0072】

次に、HTTPリクエストに対する通信応答としてHTTPサーバ12からHTTPレスポンスを受信する(S15)。そして、受信したHTTPレスポンスのHTTPボディを各パートに分割する(S16)。ここで、各パートへの分割は、「MIME\_boundary」で区分された要素に分割することであり、またここで全てのパートに関して分割する。

そしてその後、分割して得た全てのパートを順に対象として、ステップS17乃至S19の処理を繰り返す。この処理においては、まず対象のパートがサーバコマンドを記載したパートか否か判断する(S17)。そして、サーバコマンドであればサーバコマンド登録処理を行う(S18)。また、サーバコマンドでないときは、クライアントコマンドに対する応答が記載されたパートであるので、応答通知処理を行う(S19)。

#### 【0073】

ステップS18又はS19の後は、ステップS17に戻り、次のパートを対象として処理を繰り返す。そして、全てのパートについてこれらの処理を行った時点で、図17のフローチャートに示す処理を終了する。

ここまでの処理において、ステップS15及びS16ではCPU31はHTTPレスポ

ンス受信手段47として機能し、ステップS17乃至S19では受信メッセージ分配手段48として機能する。

【0074】

次に、図17のフローチャートに示した処理について、一部分ずつより詳細に示したフローチャートを用いて説明する。

図18は、図17のステップS11乃至S14の部分の処理をより詳細に示したフローチャートである。

【0075】

この処理においては、HTTPクライアント11のCPU31はまず、クライアントコマンドプール41から、「状態」が「未送信」であるクライアントコマンドシートの「メソッド名」と「入力パラメータ」の内容を、送信すべきクライアントコマンドとして収集し、「コマンドID」の内容もそのコマンドのコマンドIDとして収集する(S21)。「未送信」という「状態」は、コマンドがクライアントコマンド生成手段43によって生成された後、まだHTTPサーバ12に通知されていないことを示すものであるので、これを基準にHTTPサーバ12に送信すべきコマンドを抽出できる。

【0076】

その後、ステップS21で収集した全てのクライアントコマンドを順次対象として、ステップS22乃至S24の処理を繰り返す。これらの処理においては、まず対象のクライアントコマンドとそのコマンドIDとを、これらの情報がそれぞれSOAPボディとSOAPヘッダとに含まれるXML文書に変換し(S22)、対象のコマンドに関するパートとなるSOAPエンベロープを生成する(S23)。そして、対象のクライアントコマンドを記載していたクライアントコマンドシートの「状態」を「応答待ち」に変更する(S24)。「応答待ち」という「状態」は、コマンドをHTTPサーバ12に通知済であることを示すものである。

【0077】

これらが全て完了した後、CPU31は、サーバコマンドプール42から、「状態」が「処理完了」であるサーバコマンドシートの「出力パラメータ」の内容を、サーバコマンドに対するコマンド応答のうち送信すべきものとして収集し、「コマンドID」の内容も、対応するサーバコマンドのコマンドIDとして収集する(S25)。「処理完了」という「状態」は、サーバコマンドに対応する処理がサーバコマンド実行結果生成手段44によって生成された後、まだHTTPサーバ12に通知されていないことを示すものであるので、これを基準にHTTPサーバ12に送信すべきコマンド応答を抽出できる。

【0078】

その後、ステップS25で収集した全てのコマンド応答を順次対象として、ステップS26乃至S28の処理を繰り返す。これらの処理は、まず対象のコマンド応答とその応答と共に収集したコマンドIDとを、これらの情報がそれぞれSOAPボディとSOAPヘッダとに含まれるXML文書に変換し(S26)、対象のコマンド応答に関するパートとなるSOAPエンベロープを生成する(S27)処理である。これらの処理は、対象が異なる点以外はステップS22及びS23の処理と同じものである。そして、次に対象のコマンド応答を記載していたサーバコマンドシートの「状態」を「応答済」に変更する(S28)。「応答済」という「状態」は、コマンド応答をHTTPサーバ12に通知済であることを示すものである。

【0079】

そして、ここまでの処理が全て完了した後、CPU31は、ステップS23又はS27で生成した各パートをマージし、図11に示したようなマルチパートのHTTPリクエストを生成してHTTPサーバ12に送信する(S29)。

なお、ステップS24又はS28で行った「状態」の変更は、実際にこの送信が終了してから行うようにしてもよい。このようにすることにより、通信エラーが発生しても、送信しようとしていたコマンド及びコマンド応答を再度送信の対象とすることができるので、システムの信頼性が向上する。



以上でH T T Pリクエストの送信に関する処理を終了し、図17のステップS15以降に相当する処理に進む。

#### 【0080】

図19は、図17のステップS15以下の部分の処理をより詳細に示すフローチャートである。図18のステップS29の次の処理は、この図ではステップS31に該当する。

この処理においては、H T T Pクライアント11のC P U 31はまず、送信したH T T Pリクエストに対するH T T Pレスポンスの受信を待ち、H T T Pサーバ12からこれを受信する（S31）。これを受信すると、そのH T T Pボディを解析して各パートに分割する（S32）。

そしてその後、分割して得た各パートを順次対象として、ステップS33乃至ステップS39の処理を繰り返す。

#### 【0081】

この部分の処理においては、まず、対象のパートがサーバコマンドであるか否か判断する（S33）。上述したように、H T T Pレスポンスには、サーバコマンドと、クライアントコマンドに対する応答とが含まれている可能性があるので、対象のパートがこのいずれであるかを判断するのである。そして、この判断は、対象のパートにSOAPActionヘッダが存在するか否か、あるいはX-SOAP-Typeヘッダの内容によって判断することができる。

#### 【0082】

ステップS33でサーバコマンドでなければ、そのパートはクライアントコマンドに対する応答であるので、そのパートのXML文書を解析してクライアントコマンドシートに登録できる形式のデータに変換し（S34）、クライアントコマンドプール41からそのコマンド応答に対応するクライアントコマンドを探索し、そのクライアントコマンドについてのクライアントコマンドシートの「出力パラメータ」の項目にコマンド応答のデータを登録する（S35）。なお、コマンド応答には、「コマンドID」の情報として、クライアントコマンドの送信時に付したものと同一コマンドIDが付してあるものとし、クライアントコマンドの探索は、この情報をキーとして行うことができる。

#### 【0083】

データの登録が終わると、データを登録したクライアントコマンドシートの「状態」を「応答受信済」に変更してその旨を示す（S36）。そして、「クライアントコマンド実行結果の通知先」に登録されている通知先に、応答があった旨を通知する（S37）。この通知によって、クライアントコマンドを生成したアプリケーション等は、その生成したコマンドに応答があったことを認識し、応答に応じた処理を行うことができる。

#### 【0084】

例えば、異常通知を発するアプリケーションがH T T Pサーバ12に異常通知を行う旨のクライアントコマンドを生成した場合、このコマンドがH T T Pサーバ12に送信されると、H T T Pサーバ12はこれを正しく受け取った旨のコマンド応答を返してくる。そして、H T T Pクライアント11側では、このコマンド応答を受信すると、ここに含まれるコマンドIDを基にどのクライアントコマンドに対する応答であるかを探索し、見つかったクライアントコマンドと対応させてそのコマンド応答を登録する。そして、そのコマンドの実行結果通知先として登録されている、異常通知を発するアプリケーションに、応答があった旨を通知するのである。アプリケーションは、この通知を受けた場合にクライアントコマンドシートを参照すれば、生成したコマンドの実行結果を「出力パラメータ」の項目から取得することができる。

以上のステップS37までの処理が終了すると、次のパートがあればそれを対象としてステップS33からの処理を繰り返す。

#### 【0085】

一方、ステップS33でサーバコマンドであれば、そのパートのXML文書を解析してサーバコマンドシートに登録できる形式のデータに変換し（S38）、そのサーバコマンドに対応するサーバコマンドシートを作成して、コマンドIDと共にサーバコマンドプールに登録する（S39）。ここで、サーバコマンドの内容はサーバコマンドシートの「メ



ソッド名」及び「入力パラメータ」の項目に登録し、パートに記載されていたコマンド ID は「コマンド ID」の項目に登録する。また、「サーバコマンドの通知先」の項目には、「メソッド名」に記憶させたメソッドを実行させるアプリケーション等への参照情報を、予め用意してあるメソッドとアプリケーション等との対応関係の情報を参照して登録する。「状態」の初期値は「未処理」であり、「出力パラメータ」の初期値は NULL である。

#### 【0086】

以上のステップ S 3 9 までの処理が終了すると、次のパートがあればそれを対象としてステップ S 3 3 からの処理を繰り返す。

全てのパートについてステップ S 3 3 乃至 S 3 9 の処理が終了すると、図 1 9 のフローチャートに示した処理は終了する。

以上のような処理を行うことにより、HTTP クライアント 1 1 が、HTTP サーバ 1 2 に送信すべき動作要求と HTTP サーバ 1 2 から受信した動作要求に対する動作応答とを一括して HTTP サーバ 1 2 に送信することができる。また、HTTP サーバ 1 2 からの動作要求と HTTP サーバ 1 2 に送信した動作要求に対する動作応答とを一括して HTTP サーバ 1 2 から受信して処理することができる。

#### 【0087】

なお、ここでは送信すべき全てのパートを全て生成してからマージして送信を行うようにし、また全てのパートを受信してからこれを各パートに分割して処理を行うように説明したが、このようにする必要はない。

送信については、まず始めに HTTP ヘッダを送信し、以後パートを生成するたびにそのパートを順次送信し、全てのパートの送信が完了した時点でその旨のデータを送信するようにしてもよい。このようにしても、これらの課程で送信されるデータが 1 つのみの HTTP ヘッダを持つ論理的に連続した 1 つの HTTP リクエストであれば、1 回のセッションで転送でき、ネゴシエーションの処理は 1 回で済むので、マージして送信する場合と同様な効果を得ることができる。また、送信すべきデータのバッファに必要なメモリ容量を低減できるので、低コストの通信装置で大きなデータを取り扱うことができる。

また、受信側でも、各パートに関する処理を、各パートを受信するたびに順次行うようにすることができる。このようにした場合に容量を低減できることは、送信側の場合と同様である。

#### 【0088】

次に、サーバコマンドの実行に関する処理について説明する。

図 2 0 は、この処理の一例を示すフローチャートである。

サーバコマンドの実行に関する処理としては、まず、図 2 0 のフローチャートに示す処理を、図 1 9 に示したステップ S 3 9 の処理の後に、すなわちサーバコマンドをサーバコマンドプール 4 2 に登録した直後に行うことが考えられる。この処理において、HTTP クライアント 1 1 の CPU 3 1 は、サーバコマンド実行結果生成手段 4 4 として機能する。

#### 【0089】

そして、この処理においては、まず登録したサーバコマンドについてのサーバコマンドシートの「サーバコマンドの通知先」の情報に基づいてアプリケーション等を呼び出し、「メソッド名」や「入力パラメータ」のデータを渡してサーバコマンドに関する処理を実行させる (S 4 1)。なお、サーバコマンドに関する処理は、このフローチャートには示していないが、CPU 3 1 が別途実行することになる。

そして、これが完了すると、実行結果をサーバコマンドシートの「出力パラメータ」の項目に登録する (S 4 2) と共に、サーバコマンドシートの「状態」を「処理完了」に変更し、処理が完了したことを示して (S 4 3)、もとの図 1 9 の処理に戻る。

以上の処理を行うことにより、サーバコマンドを実行し、その結果をコマンド応答として HTTP サーバ 1 2 に送信可能な状態にすることができる。

#### 【0090】

また、サーバコマンドの実行に関する処理としては、図19に示した処理とは独立に、図21に示す処理を実行することも考えられる。この処理においても、HTTPクライアント11のCPU31は、サーバコマンド実行結果生成手段44として機能する。

この場合、CPU31は適当なタイミングで図21のフローチャートに示す処理を開始する。

#### 【0091】

そして、この処理においては、まずサーバコマンドプールに「状態」が「処理待ち」であるサーバコマンドシートがあるか否か判断し(SX1)、なければこのようなサーバコマンドシートが追加されるまで待機する。そして、このようなサーバコマンドシートを発見した場合、その1つを処理対象とし、そのサーバコマンドシートの「状態」を「処理中」に変更する(SX2)。

その後、図20の場合と同様なステップS41乃至S43の処理を行って処理対象のサーバコマンドシートに記載されたサーバコマンドを実行し、これが完了するとステップSX1に戻って処理を繰り返す。

#### 【0092】

以上の処理は、複数のスレッド(例えば4スレッド)で同時に行うようにしてもよい。処理対象となったサーバコマンドシートの「状態」は、「処理待ち」ではないため、複数のスレッドで同時に処理を行っても、1つのサーバコマンドシートを重複して処理対象としてしまうことはない。

#### 【0093】

以上のような処理を行うようにすれば、任意のタイミングでサーバコマンドを実行することができるので、実行に時間のかかるコマンドがあった場合でも、以後の処理が滞ることがない。そして、実行の終了したものから順に、その結果をコマンド応答としてHTTPサーバ12に送信可能な状態にすることができる。

以上で、HTTPクライアント11において実行する、各コマンド及びコマンド応答の転送に関する処理の説明を終了する。

#### 【0094】

次に、HTTPサーバ12側の機能構成について説明する。ハードウェアについては、図7を用いて説明した通り、HTTPクライアント11と同様なものを用いることができる。従って、以下のHTTPサーバ12の説明において、ハードウェアの構成要素を指す場合には、HTTPクライアント11の場合と共通の符号を用いるものとする。

図22は、HTTPサーバ12の機能のうち、コマンド及びコマンド応答に関する処理を行うための機能の構成を示す機能ブロック図である。

図22に示す機能のうち、サーバコマンドプール141及びクライアントコマンドプール142は、いずれかの書き換え可能な記憶手段に設けられるものである。例えばSDカード34に設けることができるが、RAM33や図示しないHDD(ハードディスクドライブ)に設けてもよい。サーバコマンド生成手段143、クライアントコマンド実行結果生成手段144、送信メッセージ収集手段145、受信メッセージ分配手段148の機能は、CPU31によって実現されるものである。また、HTTPレスポンス送信手段146及びHTTPリクエスト受信手段147の機能は、CPU31及びNIC35によって実現されるものである。

#### 【0095】

これらの機能についてさらに詳述する。

まず、サーバコマンドプール141は、HTTPサーバ12に設けた第2の記憶領域に該当し、サーバコマンドと、このコマンドに対する応答と、このコマンドの識別情報とを関連付けて登録するプールである。また、クライアントコマンドプール142は、HTTPサーバ12に設けた第1の記憶領域に該当し、クライアントコマンドと、このコマンドに対する応答と、このコマンドの識別情報とを関連付けて登録するプールである。また、これらのプールを設けた記憶手段がそれぞれHTTPサーバ12の第2、第1の記憶手段に該当するものとする。

## 【0096】

サーバコマンド生成手段143は、要求生成手段に該当する。そして、サーバコマンドを生成し、このコマンドを識別する識別情報（ID）を割り当て、さらにこのコマンドを管理するための管理情報を付し、これらの情報を関連付けてテーブル形式のサーバコマンドシートとしてサーバコマンドプール141に登録する機能を有する。このうち、サーバコマンドを生成する部分には、例えばHTTPサーバ12に備えるアプリケーションが該当する。また、サーバコマンド生成手段143に、HTTPクライアント11に各コマンドを実行させる際の優先順位を生成したサーバコマンドに付する機能を設けてもよい。

## 【0097】

ここで、図23にHTTPサーバ12のサーバコマンドシートにおけるデータ構造の例を示す。

この図に示すように、HTTPサーバ12のサーバコマンドシートには、このサーバコマンドシートに記載するコマンドがサーバコマンドであり、その送信先やコマンド応答の送信元がHTTPクライアント11である点で若干異なるが、図9に示したHTTPクライアント11のクライアントコマンドシートとはほぼ同様なデータを記憶する領域を設けている。「サーバコマンド実行結果の通知先」については、このシートに記載しているコマンドがサーバコマンドであることに伴って項目の名称が異なるが、内容は図9の「クライアントコマンド実行結果の通知先」と同様なものである。

## 【0098】

図22の説明に戻ると、クライアントコマンド実行結果生成手段144は、応答生成手段に該当する。そして、クライアントコマンドプール142からクライアントコマンドを読み出して実行するアプリケーションである。そして、クライアントコマンドに対する応答を生成し、クライアントコマンドのコマンドIDと関連付けてクライアントコマンドプール142に登録する機能を有する。なお、HTTPクライアント11から受信したクライアントコマンドは、このコマンドを識別するID及びこのコマンドを管理するための管理情報と関連付けて、テーブル形式のクライアントコマンドシートとしてクライアントコマンドプール142に登録しておくようにしている。そして、クライアントコマンド実行結果生成手段144が生成したコマンド応答も、実行したクライアントコマンドについてのクライアントコマンドシートに登録する。

## 【0099】

また、クライアントコマンド実行結果生成手段144に、クライアントコマンドプール142から複数の種類のクライアントコマンドを読み出し、各クライアントコマンドに対する応答を生成する機能を設けることが考えられる。さらに、クライアントコマンドがHTTPサーバ12に優先して処理を実行させるための実行優先順位の情報を含む場合には、優先順位の高いものから優先的に読み出して実行する機能を設けることも考えられる。

なお、クライアントコマンド実行結果生成手段144は、アプリケーションそのものではなく、クライアントコマンドの実行に必要なアプリケーションを呼び出してコマンドを実行させるモジュールであってもよい。

## 【0100】

ここで、図24にHTTPサーバ12のクライアントコマンドシートにおけるデータ構造の例を示す。

この図に示すように、HTTPサーバ12のクライアントコマンドシートには、このクライアントコマンドシートに記載するコマンドがクライアントコマンドであり、その送信元やコマンド応答の送信先がHTTPクライアント11である点で若干異なるが、図10に示したHTTPクライアント11のサーバコマンドシートとはほぼ同様なデータを記憶する領域を設けている。「クライアントコマンドの通知先」については、このシートに記載しているコマンドがサーバコマンドであることに伴って項目の名称が異なるが、内容は図10の「サーバコマンドの通知先」と同様なものである。

## 【0101】

再び図22の説明に戻ると、送信メッセージ収集手段145は、収集手段に該当する。



そして、クライアントコマンド実行結果生成手段 1 4 4 が生成したコマンド応答とこのコマンド応答に対応するクライアントコマンドのコマンド ID とを関連付けてクライアントコマンドプール 4 2 から読み出すと共に、サーバコマンド生成手段 1 4 3 が生成したサーバコマンドとこのコマンドのコマンド ID とを関連付けてサーバコマンドプール 1 4 1 から読み出し、これらから送信メッセージを生成する機能を有する。

なお、コマンド応答やサーバコマンドに実行優先順位が指定されている場合には、送信メッセージ収集手段 1 4 5 がそれぞれ実行優先順位の高いものから順に読み出すようにすることが考えられる。

送信メッセージの形式については、HTTP クライアント 1 1 の場合と同様である。

#### 【0 1 0 2】

また、HTTP レスポンス送信手段 1 4 6 は、送信手段に該当し、送信メッセージ収集手段 1 4 5 が生成した送信メッセージを含む HTTP レスポンスを、HTTP クライアント 1 1 から受信した HTTP リクエストに対する通信応答として生成し、HTTP クライアント 1 1 に送信する機能を有する。このとき、1 つの HTTP レスポンスに送信メッセージをいくつ含めてもよいし、コマンド応答に係る送信メッセージとサーバコマンドに係る送信メッセージとを任意に混在させることもできる。

そこで、HTTP レスポンス送信手段 1 4 6 は、これらのいずれに係る送信メッセージに関わり無く、送信メッセージ収集手段 1 4 5 が生成した全ての送信メッセージを 1 つの HTTP レスポンスに含めて送信するようにしている。ただし、1 つの HTTP レスポンスに含める送信メッセージの数に上限を設けることも考えられる。

#### 【0 1 0 3】

ところで、HTTP レスポンスの送信は、送信メッセージ収集手段 1 4 5 がサーバコマンドやコマンド応答等の読み出しを試みた場合には、読み出すデータがなく、結果的に送信すべき SOAP メッセージを生成しなかった場合にも行うものである。そして、この読み出しの試みは、HTTP クライアント 1 1 からの HTTP リクエストを受信した場合に行うものとする。

このようにするのは、上述のように、HTTP サーバ 1 2 からファイアウォール 1 4 を越えて HTTP クライアント 1 1 に HTTP リクエストを送信することができないためである。

#### 【0 1 0 4】

HTTP リクエスト受信手段 1 4 7 は、受信手段に該当し、HTTP クライアント 1 1 からの HTTP リクエストを受信する機能を有する。そしてここでは、HTTP リクエストには、クライアントコマンド及びそのコマンドと関連付けられたコマンド ID を含む受信メッセージと、サーバコマンドに対する応答及びそのコマンドと関連付けられたコマンド ID を含む受信メッセージとが、任意に混在して含まれている。

ここで、受信メッセージとは、上記のコマンドや応答とコマンド ID とを SOAP メッセージとして記載したものである。

#### 【0 1 0 5】

受信メッセージ分配手段 1 4 8 は、分配手段に該当する。そして、HTTP リクエスト受信手段 1 4 7 が受信した HTTP リクエストに含まれるデータを、サーバコマンドプール 1 4 1 及びクライアントコマンドプール 1 4 2 に振り分けて登録する機能を有する。

具体的には、クライアントコマンド及びそのコマンドと関連付けられたコマンド ID とをクライアントコマンドプール 1 4 2 にクライアントコマンドシートを設けて登録すると共に、サーバコマンドに対する応答については、そのコマンドと関連付けられたコマンド ID をサーバコマンドプール 1 4 1 に記憶しているサーバコマンドシートのコマンド ID と照合して対応するサーバコマンドを特定し、そのサーバコマンドについての「出力パラメータ」として登録する。

#### 【0 1 0 6】

そしてこのとき、HTTP リクエストを分割してそこに含まれる各受信メッセージを取り出し、そのデータをテーブルへの登録に必要な形式に変換するが、この変換は、WSD

Lに基づいて生成される所要の変換プログラム（デシリアライザ）を実行することによって行うことができる。

このような機能を有するHTTPサーバ12が受信するHTTPリクエストは、HTTPクライアント11から送信されてくるものである。例えばHTTPクライアント11の機能の説明中で図11を用いて説明したものである。HTTPサーバ12が送信するHTTPレスポンスも、HTTPクライアント11に対して送信し、HTTPクライアント11が受信するものである。例えば図12を用いて説明したものである。これらに含まれるパートの内容も、図13乃至図16を用いて説明したようなものとなる。

#### 【0107】

次に、以上説明したような構成及び機能を有するHTTPサーバ12において実行する処理について、図25乃至図29のフローチャートを用いて説明する。これらのフローチャートに示す処理は、HTTPサーバ12のCPU31が所要の制御プログラムを実行することによって行うものである。

#### 【0108】

まず、図25にメッセージの収集及び分配処理の基本動作のフローチャートを示す。

HTTPサーバ12のCPU31は、HTTPクライアント11からHTTPリクエストが送信されてくると、図25のフローチャートに示す処理を開始する。

そして、まずそのHTTPリクエストを受信する（S111）。そして、受信したHTTPリクエストのHTTPボディを各パートに分割する（S112）。ここで、各パートへの分割は、「MIME\_boundary」で区分された要素に分割することであり、またここで全てのパートに関して分割する。

#### 【0109】

そしてその後、分割して得た全てのパートを順に対象として、ステップS113乃至S115の処理を繰り返す。この処理においては、まず対象のパートがクライアントコマンドを記載したパートか否か判断する（S113）。そして、クライアントコマンドであればクライアントコマンド登録処理を行う（S114）。また、クライアントコマンドでないときは、サーバコマンドに対する応答が記載されたパートであるので、応答通知処理を行う（S115）。

#### 【0110】

ステップS114又はS115の後は、ステップS113に戻り、次のパートを対象として処理を繰り返す。そして、全てのパートについてこれらの処理を行った時点で、次のステップS116に進む。

ここまでの処理において、ステップS111及びS112ではCPU31はHTTPリクエスト受信手段147として機能し、ステップS113乃至S115では受信メッセージ分配手段148として機能する。

#### 【0111】

次に、CPU31はサーバコマンドの収集処理を行う（S116）。この処理は、サーバコマンドプール41からHTTPクライアント11に送信すべきサーバコマンドを収集する処理であり、収集したデータからSOAPエンベロープによるパートを生成する処理を含む。

次に、クライアントコマンドに対する応答であるクライアントコマンド実行結果の収集処理を行う（S117）。この処理は、クライアントコマンドプールからHTTPクライアント11に送信すべきコマンド応答を収集する処理であり、やはり収集したデータからSOAPエンベロープによるパートを生成する処理を含む。

#### 【0112】

その後、ステップS116及びS117の処理で生成したパートを1つにマージして、すべてのパートを含むHTTPレスポンスを生成し（S118）、そのHTTPレスポンスを、ステップS111で受信したHTTPリクエストに対する通信応答としてHTTPクライアント11に送信して（S119）処理を終了する。

ここまでの処理において、ステップS116及びS117ではCPU31は送信メッセ

ージ収集手段145として機能し、ステップS118及びS119ではHTTPレスポンス送信手段146として機能する。

#### 【0113】

次に、図25のフローチャートに示した処理について、一部分ずつより詳細に示したフローチャートを用いて説明する。

図26は、図25のステップS111乃至S115の部分の処理をより詳細に示したフローチャートである。

この処理においては、HTTPサーバ12のCPU31はまず、HTTPクライアントから送信されてきたHTTPリクエストを受信する(S121)。そして、これを受信すると、そのHTTPボディを解析して各パートに分割する(S122)。

そしてその後、分割して得た各パートを順次対象として、ステップS123乃至ステップS129の処理を繰り返す。

#### 【0114】

この部分の処理においては、まず、対象のパートがクライアントコマンドであるか否か判断する(S123)。上述したように、HTTPリクエストには、クライアントコマンドと、サーバコマンドに対する応答とが含まれている可能性があるので、対象のパートがこのいずれであるかを判断するのである。そして、この判断は、対象のパートにSOAPActionヘッダが存在するか否か、あるいはX-SOAP-Typeヘッダの内容によって判断することができる。

#### 【0115】

ステップS123でクライアントコマンドでなければ、そのパートはサーバコマンドに対する応答のパートであるので、そのパートのXML文書を解析してサーバコマンドシートに登録できる形式のデータに変換し(S124)、サーバコマンドプール141からそのコマンド応答に対応するサーバコマンドを探索し、そのサーバコマンドについてのサーバコマンドシートの「出力パラメータ」の項目にコマンド応答のデータを登録する(S125)。なお、コマンド応答には、「コマンドID」の情報として、サーバコマンドの送信時に付したものと同一コマンドIDが付してあるものとし、サーバコマンドの探索は、この情報をキーとして行うことができる。

#### 【0116】

データの登録が終わると、データを登録したサーバコマンドシートの「状態」を「応答受信済」に変更してその旨を示す(S126)。そして、「サーバコマンド実行結果の通知先」に登録されている通知先に、応答があった旨を通知する(S127)。この通知によって、サーバコマンドを生成したアプリケーション等は、その生成したコマンドに応答があったことを認識し、応答に応じた処理を行うことができる。

#### 【0117】

例えば、HTTPクライアント11での異常発生に対応するアプリケーションがHTTPクライアント11の温度センサのセンサ値を取得するサーバコマンドを生成した場合、このコマンドがHTTPクライアント11に送信されると、HTTPクライアント11は要求されたセンサ値を含むコマンド応答を返してくる。そして、HTTPサーバ12側では、このコマンド応答を受信すると、ここに含まれるコマンドIDを基にどのサーバコマンドに対する応答であるかを探索し、見つかったサーバコマンドと対応させてそのコマンド応答を登録する。そして、そのコマンドの実行結果通知先として登録されている、異常発生に対応するアプリケーションに、応答があった旨を通知するのである。アプリケーションは、この通知を受けた場合にサーバコマンドシートを参照すれば、生成したコマンドの実行結果を「出力パラメータ」の項目から取得することができる。

以上のステップS127までの処理が終了すると、次のパートがあればそれを対象としてステップS123からの処理を繰り返す。

#### 【0118】

一方、ステップS123でクライアントコマンドであれば、そのパートのXML文書を解析してクライアントコマンドシートに登録できる形式のデータに変換し(S128)、



そのクライアントコマンドに対応するクライアントコマンドシートを作成して、コマンドIDと共にクライアントコマンドプールに登録する(S129)。ここで、クライアントコマンドの内容はクライアントコマンドシートの「メソッド名」及び「入力パラメータ」の項目に登録し、パートに記載されていたコマンドIDは「コマンドID」の項目に登録する。また、「クライアントコマンドの通知先」の項目には、「メソッド名」に記憶させたメソッドを実行させるアプリケーション等への参照情報を、予め用意してあるメソッドとアプリケーション等との対応関係の情報を参照して登録する。「状態」の初期値は「未処理」であり、「出力パラメータ」の初期値はNULLである。

#### 【0119】

以上のステップS129までの処理が終了すると、次のパートがあればそれを対象としてステップS123からの処理を繰り返す。

全てのパートについてステップS123乃至S129の処理が終了すると、図26のフローチャートに示した処理は終了する。

以上でHTTPリクエストの受信に関する処理を終了し、図25のステップS116以降に相当する処理に進む。

#### 【0120】

図27は、図25のステップS116以降の部分の処理をより詳細に示すフローチャートである。図26のステップS127又はS129の次の処理は、この図ではステップS131に該当する。

この処理においては、HTTPサーバ12のCPU31はまず、サーバコマンドプール141から、「状態」が「未送信」であるサーバコマンドシートの「メソッド名」と「入力パラメータ」の内容を、送信すべきサーバコマンドとして収集し、「コマンドID」の内容もそのコマンドのコマンドIDとして収集する(S131)。HTTPサーバ12においては、「未送信」という「状態」は、コマンドがサーバコマンド生成手段143によって生成された後、まだHTTPクライアント11に通知されていないことを示すものであるので、これを基準にHTTPクライアント11に送信すべきコマンドを抽出できる。

#### 【0121】

その後、ステップS131で収集した全てのサーバコマンドを順次対象として、ステップS132乃至S134の処理を繰り返す。これらの処理においては、まず対象のサーバコマンドとそのコマンドIDとを、これらの情報がそれぞれSOAPボディとSOAPヘッダとに含まれるXML文書に変換し(S132)、対象のコマンドに関するパートとなるSOAPエンベロープを生成する(S133)。そして、対象のサーバコマンドを記載していたサーバコマンドシートの「状態」を「応答待ち」に変更する(S134)。ここでの「応答待ち」という「状態」は、コマンドをHTTPクライアント11に通知済であることを示すものである。

#### 【0122】

これらが全て完了した後、CPU31は、クライアントコマンドプール142から、「状態」が「処理完了」であるクライアントコマンドシートの「出力パラメータ」の内容を、クライアントコマンドに対するコマンド応答のうち送信すべきものとして収集し、「コマンドID」の内容も、対応するクライアントコマンドのコマンドIDとして収集する(S135)。HTTPサーバ12においては、「処理完了」という「状態」は、クライアントコマンドに対応する処理がクライアントコマンド実行結果生成手段144によって生成された後、まだHTTPクライアント11に通知されていないことを示すものであるので、これを基準にHTTPクライアント11に送信すべきコマンド応答を抽出できる。

#### 【0123】

その後、ステップS135で収集した全てのコマンド応答を順次対象として、ステップS136乃至S138の処理を繰り返す。これらの処理は、まず対象のコマンド応答とその応答と共に収集したコマンドIDとを、これらの情報がSOAPボディとSOAPヘッダとにそれぞれ含まれるXML文書に変換し(S136)、対象のコマンド応答に関するパートとなるSOAPエンベロープを生成する(S137)処理である。これらの処理は

、対象が異なる点以外はステップS 1 3 2 及びS 1 3 3 の処理と同じものである。そして、次に対象のコマンド応答を記載していたクライアントコマンドシートの「状態」を「応答済」に変更する（S 1 3 8）。「応答済」という「状態」は、ここではコマンド応答をH T T Pクライアント1 1 に通知済であることを示すものである。

#### 【0 1 2 4】

そして、ここまでの処理が全て完了した後、C P U 3 1 は、ステップS 1 3 3 又はS 1 3 7 で生成した各パートをマージし、図1 2 に示したようなマルチパートのH T T Pレスポンスを生成してH T T Pクライアント1 1 に送信する。

なお、ステップS 1 3 4 又はS 1 3 8 で行った「状態」の変更は、実際にこの送信が終了してから行うようにしてもよい。このようにすることにより、通信エラーが発生しても、送信しようとしていたコマンド及びコマンド応答を再度送信の対象とすることができるので、システムの信頼性が向上する。

#### 【0 1 2 5】

なお、ここでは送信すべきパートを全て生成してからマージして送信を行うようにし、また全てのパートを受信してからこれを各パートに分割して処理を行うように説明したが、このようにする必要はない。各パートを生成するたびに順次送信するようしたり、各パートを受信するたびに順次各パートに関する処理を行うようにしてもよいことは、H T T Pクライアント1 1 の場合と同様である。

#### 【0 1 2 6】

次に、クライアントコマンドの実行に関する処理について説明する。

図2 8 は、この処理の一例を示すフローチャートである。

クライアントコマンドの実行に関する処理としては、まず、図2 8 のフローチャートに示す処理を、図2 7 に示したステップS 1 2 9 の処理の後に、すなわちクライアントコマンドをクライアントコマンドプール4 2 に登録した直後に行うことが考えられる。この処理において、H T T Pサーバ1 2 のC P U 3 1 は、クライアントコマンド実行結果生成手段1 4 4 として機能する。

#### 【0 1 2 7】

そして、この処理においては、まず登録したクライアントコマンドについてのクライアントコマンドシートの「クライアントコマンドの通知先」の情報に基づいてアプリケーション等呼び出し、「メソッド名」や「入力パラメータ」のデータを渡してクライアントコマンドに関する処理を実行させる（S 1 4 1）。なお、クライアントコマンドに関する処理は、このフローチャートには示していないが、C P U 3 1 が別途実行することになる。

#### 【0 1 2 8】

そして、これが完了すると、実行結果をクライアントコマンドシートの「出力パラメータ」の項目に登録する（S 1 4 2）と共に、クライアントコマンドシートの「状態」を「処理完了」に変更し、処理が完了したことを示して（S 1 4 3）、もとの図2 7 の処理に戻る。

以上の処理を行うことにより、クライアントコマンドを実行し、その結果をコマンド応答としてH T T Pクライアント1 1 に送信可能な状態にすることができる。

#### 【0 1 2 9】

また、クライアントコマンドの実行に関する処理としては、図2 7 に示した処理とは独立に、図2 9 に示す処理を実行することも考えられる。この処理においても、H T T Pサーバ1 2 のC P U 3 1 は、クライアントコマンド実行結果生成手段1 4 4 として機能する。

この場合、C P U 3 1 は適当なタイミングで図2 9 のフローチャートに示す処理を開始する。

#### 【0 1 3 0】

そして、この処理においては、まずクライアントコマンドプールに「状態」が「処理待ち」であるクライアントコマンドシートがあるか否か判断し（S Y 1）、なければこのよ

うなクライアントコマンドシートが追加されるまで待機する。そして、このようなクライアントコマンドシートを発見した場合、その1つを処理対象とし、そのクライアントコマンドシートの「状態」を「処理中」に変更する（S Y 2）。

その後、図 2 8 の場合と同様なステップ S 1 4 1 乃至 S 1 4 3 の処理を行って処理対象のクライアントコマンドシートに記載されたクライアントコマンドを実行し、これが完了するとステップ S Y 1 に戻って処理を繰り返す。

以上の処理を複数のスレッドで同時に行うようにしてもよいことは、H T T P クライアント 1 1 の場合と同様である。

#### 【 0 1 3 1 】

以上のような処理を行うようにすれば、任意のタイミングでクライアントコマンドを実行することができるので、実行に時間のかかるコマンドがあった場合でも、以後の処理が滞ることがない。そして、実行の終了したものから順に、その結果をコマンド応答として H T T P クライアント 1 1 に送信可能な状態にすることができる。

以上で、H T T P サーバ 1 2 において実行する、各コマンド及びコマンド応答の転送に関する処理の説明を終了する。

#### 【 0 1 3 2 】

H T T P クライアント 1 1 と H T T P サーバ 1 2 とに以上説明してきたような各機能を受け、各処理を行わせることにより、送信元から通信相手に送信すべき動作要求と、通信相手から受信した動作要求に対する動作応答とを、一括して通信相手に送信することができるので、動作要求の送信と動作応答の送信とについて別々にネゴシエーションを行って通信のコネクションを確立する必要がなく、通信のオーバーヘッドを低減して通信効率を高めることができる。

#### 【 0 1 3 3 】

また、動作要求と動作応答とを一括して送信できるのは、これらをそれぞれ直列化したデータに変換し、構造化言語形式で記載した送信メッセージに変換しているためである。このようにしたことにより、フォーマットの異なる動作要求と動作応答とを容易に結合し、論理的に1つの送信内容として送信することができるのである。

また、このようにしたことにより、受信側でも、通信相手に送信した動作要求に対する動作応答と通信相手からの動作要求とを一括して受信し、その受信した内容を容易に個々のメッセージに分離し、それが動作要求であるか動作応答であるかに応じて適切な処理を行うことができる。

#### 【 0 1 3 4 】

さらに、通信要求を常に一方の装置から発して通信を行い、その通信相手から通信要求元への動作要求等の送信は、その通信要求に対する応答として行うようにすれば、通信要求を発する側の装置（通信クライアント）がファイアウォールの内側に設けられているような通信システムであっても、ファイアウォールの存在を意識せずに動作要求及び動作応答の送受信を行うことができる。また、通信要求と通信応答とが対応しているため、通信のレベルでのタイミング管理が容易である。

#### 【 0 1 3 5 】

この場合において、上記の一方の装置から通信相手に定期的に通信要求を送信するようにすれば、ファイアウォールの外側から内側に向けての情報の送信が長時間に亘って停滞してしまう事態を防止できる。

また、クライアントコマンドプールやサーバコマンドプールを設け、各アプリケーション等が生成した動作要求や動作応答をこれらのプールに蓄積しておくようにすることにより、動作要求や動作応答の生成を、通信相手に対する送信タイミングを考慮せずに行うことができる。従って、アプリケーション等が行う処理を簡略化することができ、設計や開発が容易になる。

#### 【 0 1 3 6 】

そして、このようにした場合でも、通信相手に送信すべき動作要求や動作応答をプールから読み出す収集手段を設けることにより、通信を行う場合に、送信すべき情報を漏れな



く送信することができる。

また、受信した動作要求や動作応答を各々分離して適切なプールに記憶させる分配手段を設け、受信した情報もプールに蓄積しておくようにすることにより、受信した動作要求に係る動作の実行や、動作応答受信後の処理を、通信相手からの受信タイミングを考慮せずに行うことができる。従って、アプリケーション等が行う処理を簡略化することができ、設計や開発が容易になる。

#### 【0137】

また、生成した動作要求にID等の識別情報を付して、動作要求を記憶、送信する際にこの識別情報と関連付けて行うようにし、また動作応答を記憶、送信する際にも対応する動作要求の識別情報と関連付けて行うようにすれば、1つのメッセージ（ここではHTTPメッセージ）に複数の動作要求や動作応答を含める場合でも、その識別情報を媒介に動作要求と動作応答との対応関係を容易に認識することができる。

さらに、動作要求に優先順位を付して、これが高いものから順に実行したり応答を送信したりするようにすれば、緊急を要する動作を優先的に実行すると共にその応答も優先的に返すことができる。

#### 【0138】

〔SMTPを採用する場合の参考例：図30乃至図35〕

次に、通信プロトコルにSMTPを採用する場合の参考例について説明する。この参考例は、上述のHTTPを採用する場合の実施例と共通点が多いので、共通点については説明を省略するか簡単にし、相違点を中心に説明する。

まず、図30に、SMTPを採用する場合の参考例を適用する通信システムの構成例を示す。

#### 【0139】

この通信システムは、図30に示すように、通信装置AとメールサーバA'とを接続したLAN\_Aと、通信装置BとメールサーバB'とを接続したLAN\_Bとを、それぞれファイアウォールAとファイアウォールBとを介してインターネット13に接続して構成している。また、ファイアウォールを介して外部からアクセス可能な位置に、LAN\_A側ではメールサーバAを、LAN\_B側ではメールサーバBをそれぞれ設けている。そして、通信装置Aが第1の通信装置に、通信装置Bが第2の通信装置に該当する。

#### 【0140】

SMTPを用いて通信を行う場合には、通信装置Aと通信装置Bとの間での情報転送は電子メールによって行う。そして、例えば通信装置Bから通信装置Aに情報を送信する場合、図30に破線で示したように、通信装置Bは通信装置Aを宛先とする電子メールを送信し、これをまずメールサーバB'に送信する。すると、各メールサーバB', B, Aを順次介して、通信装置Aが直接アクセスするメールサーバA'まで電子メールが転送される。また、図示は省略したが、通常はメールサーバBとメールサーバAとの間に更に別のメールサーバを介して転送を行うことになる。

#### 【0141】

そして、一点鎖線で示したように通信装置Aが定期的にメールサーバA'にアクセスすることにより、通信装置A宛の電子メールを受け取ることができ、以上で通信装置Bから通信装置Aへの情報転送が完了する。通信装置Aから通信装置Bへの情報転送は、逆の手順で行うことができ、情報転送に関しては、通信装置Aと通信装置Bとは対称である。ただし、メールサーバA'及びB'を設けることは必須ではなく、通信装置AがメールサーバAと、通信装置BがメールサーバBと直接通信を行うようにしてもよい。また、メール受信用のメールサーバとメール送信用のメールサーバが異なってもよい。

#### 【0142】

このような通信システムにおいて、各LANには外部からアクセス可能な位置にメールサーバを設けていることから、ファイアウォールを越えて電子メールを転送することができる。

なお、この参考例においては、通信装置Aと通信装置Bとが直接ネゴシエーションをし

た上で通信を行っているわけではないが、相互に情報転送を行うことが可能である。

【0 1 4 3】

また、通信装置 A 及び通信装置 B も、図 1 に示した第 1 の通信装置及び第 2 の通信装置の場合と同様に、互いの制御管理を行うためのアプリケーションプログラムを実装している。そして、R P C により、互いの実装するアプリケーションプログラムのメソッドに対する処理の依頼である「動作要求」を送信し、この依頼された処理の結果である「動作応答」を取得することができるようになっている。

【0 1 4 4】

図 3 1 に、これらの動作要求と動作応答の関係を示すが、動作要求のレベルでは、通信装置 A と通信装置 B との関係は、既に説明した H T T P クライアント 1 1 と H T T P サーバ 1 2 との関係と同様に、対称なものである。しかし、S M T P の場合には、通信のレベルでも通信装置 A と通信装置 B とが対称な関係にあることが、H T T P の場合と異なる。

【0 1 4 5】

図 3 2 に、この通信システムにおける通信シーケンスの例を示す。

上述したように、この通信システムにおいては、通信装置 A と通信装置 B との間の通信は、電子メールを用いて行う。そして、電子メールには、送信元と宛先は存在し、その宛先から送信元に返信を行うことも可能である。しかし、初めの電子メールと返信とは全く独立したものであり、H T T P の場合のような、通信要求と通信応答のような関係はない。

従って、どちらの通信装置から先に通信を行ってもよいし、交互に電子メールを送信しなければならないということもないのであるが、ここでは、通信装置 A から通信装置 B にまず電子メールを送信し、交互に計 3 通の電子メールを送受信する場合の例を示している。

【0 1 4 6】

これらの電子メールには、送信先（宛先）に対する動作要求（コマンド）及び、送信先から受信したコマンドに対する動作応答（コマンド応答、あるいは単に「応答」）を記載して送信するようにしている。これは、通信装置 A と通信装置 B のどちらが送信元であっても同様である。

従って、例えば通信装置 A 側コマンド a は、電子メール x に記載して転送し、通信装置 B からのコマンド応答をその後の電子メール y に記載して転送することができる。また、通信装置 B 側コマンド c は、電子メール y に記載して転送し、通信装置 A からのコマンド応答をその後の電子メール z に記載して転送することができる。

【0 1 4 7】

なお、動作要求及び動作応答に対する応答をそれぞれ任意の数ずつ（0 でもよい）1 通の電子メールに記載することができる。そして、1 通の電子メールに記載した内容は、1 つのメッセージであり、当然論理的に一括して転送する。そして、このようにすることにより、必要な情報を転送するための電子メールの数を減らし、オーバーヘッドを低減して通信の効率化を図っている。

また、コマンドを受信した後最初に送信する電子メールにコマンド応答を記載しなくてもよいことは、図 6 を用いて説明した H T T P の場合と同様である。

【0 1 4 8】

次に、通信装置 A 及び通信装置 B における、コマンドやコマンド応答を結合し、また分離する処理を行うための機能構成及びその処理の手順について説明する。ハードウェア構成については、H T T P を用いる実施例の説明において図 7 に示した H T T P クライアント 1 1 の場合と同様なものを使用することができる。

【0 1 4 9】

図 3 3 は、通信装置 A の機能のうち、コマンド及びコマンド応答に関する処理を行うための機能の構成を示す図 8 と対応する機能ブロック図である。

図 3 3 に示す機能のうち、通信装置 A 側コマンドプール 5 1，通信装置 B 側コマンドプール 5 2，通信装置 A 側コマンド生成手段 5 3，通信装置 B 側コマンド実行結果生成手段

54は、それぞれ図8に示したクライアントコマンドプール41, サーバコマンドプール42, クライアントコマンド生成手段43, サーバコマンド実行結果生成手段44と対応する機能であり、装置の名称が異なることにより名称を変更したものである。

#### 【0150】

また、送信メッセージ収集手段45及び受信メッセージ分配手段48は、図8に示した同名の手段と対応するものである。ただし、送受信するためのデータの形式が、SMTPに対応した形式である点が図8に示した例とは異なる。しかし、コマンドやコマンド応答に対応する、個々のSOAPメッセージについては、図8の場合と同様なものである。

メール送信手段56及びメール受信手段57については、それぞれ送信手段及び受信手段に該当するが、送受信に使用するプロトコルが異なることに伴って、図8に示したHTTPリクエスト送信手段46及びHTTPレスポンス受信手段47とは異なるものである。

#### 【0151】

すなわち、メール送信手段56は、送信メッセージ収集手段45が生成した送信メッセージを含む、通信装置B宛の電子メールを生成し、メールサーバA'に送信する機能を有する。なお、それ以降の電子メールの転送には、通信装置Aは関与しない。また、1つの電子メールに送信メッセージをいくつ含めてもよいし、コマンド応答に係る送信メッセージと通信装置A側コマンドに係る送信メッセージとを任意に混在させることもできる点等は、図8に示した例の場合と同様である。

#### 【0152】

メール受信手段57は、定期的にメールサーバA'にアクセスして通信装置A宛ての新着メールの有無を調べ、新着メールがあった場合にこれを受信する機能を有する。そしてここでは、受信する電子メールには、通信装置B側コマンド及びそのコマンドと関連付けられたコマンドIDを含む受信メッセージと、通信装置A側コマンドに対する応答及びそのコマンドと関連付けられたコマンドIDを含む受信メッセージとが、任意に混在して含まれている。

#### 【0153】

次に、このような機能を有する通信装置Aが通信装置B宛てに送信する電子メールの例を図34に示す。

この電子メールは、図11に示したHTTPリクエストの場合と同様に、ボディ部としてMIME (Multipurpose Internet Mail Extension) に従ったマルチパートのメッセージが記載され、この各パートには、それぞれSOAPエンベロープが埋め込まれている。すなわち、ボディの部分はHTTPリクエストの場合と同様なものになっている。

#### 【0154】

しかし、ヘッダ部分はHTTPリクエストの場合とは異なり、電子メールの送信元アドレスを示すFrom、宛先アドレスを示すTo、表題を示すSubject等の情報が記載されている。

通信装置Bから通信装置Aに送信する電子メールについては、ヘッダのうちFromの内容とToの内容を入れ替えたものになる。

また、これらの電子メールに記載されるSOAPエンベロープの内容は、HTTPリクエストやHTTPレスポンスの場合と同様なものである。ただし、各パートのエンティティヘッダの部分は、データのエンコード方式が異なるためHTTPの場合とは一部が異なる。

#### 【0155】

次に、以上説明したような構成及び機能を有する通信装置Aにおいて実行する処理について、図35及び図36のフローチャートを用いて説明する。これらのフローチャートに示す処理は、通信装置AのCPUが所要の制御プログラムを実行することによって行うものである。

#### 【0156】

まず、図35にメッセージ送信時の処理の基本動作のフローチャートを示す。



通信装置 A の C P U は、送信メッセージ収集手段 4 5 が通信装置 A 側コマンドやコマンド応答等の読み出しを試みるタイミングになると、図 3 5 のフローチャートに示す処理を開始する。

そして、まず通信装置 A 側コマンドの収集処理を行う (S 5 1)。この処理は、通信装置 A 側コマンドプール 5 1 から通信装置 B に送信すべき通信装置 A 側コマンドを収集する処理であり、収集したデータから S O A P エンベロープのパートを生成する処理を含む。

#### 【0 1 5 7】

次に、通信装置 B 側コマンドに対する応答である通信装置 B 側コマンド実行結果の収集処理を行う (S 5 2)。この処理は、通信装置 B 側コマンドプール 5 2 から通信装置 B に送信すべきコマンド応答を収集する処理であり、やはり収集したデータから S O A P エンベロープのパートを生成する処理を含む。

その後、ステップ S 5 1 及び S 5 2 の処理で生成したパートを 1 つにマージして、すべてのパートを含む電子メールを生成し (S 5 3)、その電子メールを通信装置 B に宛てて送信して (S 5 4)、処理を終了する。

#### 【0 1 5 8】

以上の処理において、ステップ S 5 1 及び S 5 2 では C P U 3 1 は送信メッセージ収集手段 4 5 として機能し、ステップ S 5 3 及び S 5 4 ではメール送信手段 5 6 として機能する。

これらの処理は、図 1 7 に示したステップ S 1 1 乃至 S 1 4 の処理と対応するものであるが、S M T P の場合には、電子メールの送信と受信の間には、通信要求と通信応答のような関係はないため、処理は一旦ここで終了し、電子メールを受信する場合には、別途図 3 6 に示す処理を行う。

#### 【0 1 5 9】

図 3 6 には、メッセージ受信時の処理の基本動作のフローチャートを示す。

通信装置 A の C P U は、定期的にメールサーバ A' にアクセスし、通信装置 A 宛ての新着の電子メールがあると、図 3 6 のフローチャートに示す処理を開始する。

この処理においては、まず新着の電子メールを受信する (S 6 1)。そして、受信した電子メールのボディ (本文) を、各パートに分割する (S 6 2)。ここで、各パートへの分割は、「MIME\_boundary」で区分された要素に分割することであり、またここで全てのパートに関して分解する。

#### 【0 1 6 0】

そしてその後、全てのパートを順に対象としてステップ S 6 3 乃至 S 6 5 の処理を繰り返す。この処理においては、まず対象のパートが通信装置 B 側コマンドを記載したパートか否か判断する (S 6 3)。そして、通信装置 B 側コマンドであれば通信装置 B 側コマンド登録処理を行う (S 6 4)。また、通信装置 B 側コマンドでないときは、通信装置 A 側コマンドに対する応答が記載されたパートであるので、応答通知処理を行う (S 6 5)。

#### 【0 1 6 1】

ステップ S 6 3 又は S 6 4 の後は、ステップ S 6 3 に戻り、次のパートを対象として処理を繰り返す。そして、全てのパートについてこれらの処理を行った時点で、図 3 6 のフローチャートに示す処理を終了する。

以上の処理においては、ステップ S 6 1 では C P U 3 1 はメール受信手段 5 7 として機能し、ステップ S 6 2 乃至 S 6 5 では受信メッセージ分配手段 4 8 として機能する。

これらの処理は、図 1 7 に示したステップ S 1 5 乃至 S 1 9 の処理と対応するものである。

#### 【0 1 6 2】

また、通信装置 B 側コマンドの実行に関する処理については、図 2 0 及び図 2 1 を用いて説明したサーバコマンドの実行に関する処理と同様なものである。

以上で、通信装置 A において実行する、各コマンド及びコマンド応答の転送に関する処理の説明を終了する。

なお、通信装置 B の機能及び通信装置 B において実行する処理については、通信装置 A

側コマンドと通信装置B側コマンドとの意味合いが逆になる点以外は、通信装置Aの場合と全く同じであるので、説明を省略する。

#### 【0163】

以上説明してきたように、この発明は、SMTPのように、情報の送信と返信とが必ずしも対応しないプロトコルを使用して通信を行う場合にも適用できる。そして、この場合にも、送信元から通信相手に送信すべき動作要求と、通信相手から受信した動作要求に対する動作応答とを、一括して通信相手に送信することができるので、動作要求の送信と動作応答の送信とについて別々の単位で転送を行う必要がなく、通信のオーバーヘッドを低減して通信効率を高めることができる。

また、通信に電子メールを用いれば、ファイアウォールの内側へも容易に情報を転送することができる。

#### 【0164】

〔実施例及び参考例の変形例：図37〕

次に、上述した実施例及び参考例の変形例について説明する。

まず、上述した実施例及び参考例では、説明を簡単にするために2つの通信装置からなる通信システムを例としてこの発明について説明したが、この発明は、さらに多くの通信装置からなる通信システムやこのような通信システムを構成する通信装置に適用することも当然可能である。

例えば、HTTPを採用する場合の実施例は、図37に示すような通信システムにも適用することができる。

#### 【0165】

この通信システムは、通信機能を備えた通信装置である管理装置102と、同じく通信機能を備えた通信装置である被管理装置100とをインターネット13を含むネットワークを介して接続し、管理装置102によって被管理装置100を管理する遠隔管理システムとして構成している。

ここで、被管理装置100の具体例としては、プリンタ、スキャナ、複写機、あるいはこれらの機能を兼ね備えたデジタル複合機等の画像処理装置を始めとし、通信機能を備えたネットワーク家電、自動販売機、医療機器、電源装置、空調システム、ガス・水道・電気等の計量システム等や、ネットワークに接続可能なコンピュータ等も含め、通信機能を備える各種電子装置が考えられる。

#### 【0166】

また、この遠隔管理システムにおいては、管理装置102と被管理装置100との間の通信を仲介する通信装置である仲介装置101を設けており、管理装置102と被管理装置100とはこの仲介装置101を介して通信を行う。

そして、仲介装置101及び被管理装置100は、その利用環境に応じて多様な階層構造を成す。例えば、図37に示す設置環境Aでは、管理装置102とHTTPによる接続を確立できる仲介装置101aが、被管理装置100a及び100bを従える単純な階層構造になっているが、設置環境Bでは、4台の被管理装置100を設置するため、1台の仲介装置101を設置しただけでは負荷が大きくなる。

#### 【0167】

そのため、管理装置102とHTTPによる接続を確立できる仲介装置101bが、被管理装置100c及び100dだけでなく、他の仲介装置101cを従え、この仲介装置101cが被管理装置100e及び100fを更に従えるという階層構造を形成している。この場合、被管理装置100e及び100fを遠隔管理するために管理装置102から発せられた情報は、仲介装置101bとその下位のノードである仲介装置101cとを経由して、被管理装置100e又は100fに到達することになる。なお、各設置環境には、セキュリティ面を考慮し、ファイアウォール14を設置している。

#### 【0168】

このような遠隔管理システムにおいても、仲介装置101をHTTPクライアント、管理装置102をHTTPサーバとして取り扱うことにより、この発明を適用することがで

きる。

なお、この場合において、管理装置 1 0 2 から被管理装置 1 0 0 にコマンドを送信しようとする場合、仲介装置 1 0 1 に対して、「被管理装置 1 0 0 にコマンドを送信せよ」というコマンドを送信し、コマンドに応じた動作として被管理装置 1 0 0 に対してコマンドを送信させることが考えられる。また、宛先として被管理装置 1 0 0 のいずれかを指定して仲介装置 1 0 1 に対してコマンドを送信し、仲介装置 1 0 1 に、自己以外が宛先となっているコマンドを、その宛先に対して転送させることも考えられる。

#### 【0 1 6 9】

また、3 つ以上のノード間でコマンドやコマンド応答の送受信を行う場合、コマンドの発信元と宛先とを把握できるように、これらの情報もコマンドやコマンド応答のメッセージに含め、またコマンドシートにも記載して管理するようにするとよい。

また、メールサーバを設ければ、SMTP を採用する場合の参考例も、このような通信システムに適用することができる。

#### 【0 1 7 0】

さらに、この発明には、上記以外の変形を適用することも可能である。例えば、クライアントコマンドプール 4 1 及びサーバコマンドプール 4 2 に登録するクライアントコマンドシート及びサーバコマンドシートを、XML ドキュメントとして記載するようにしてもよい。「入力パラメータ」をデシリアライズする前の XML ドキュメントとして保存したり、「出力パラメータ」をシリアライズした後の XML ドキュメントとして保存したりしてもよい。

また、送受信するコマンドやコマンド応答の情報量に制限を設けても構わない。特に、受信するコマンドの情報量を制限するようにすると、受信側がメモリ容量の限られた装置である場合にメモリの使用量を抑えることができる。

#### 【0 1 7 1】

また、上述した実施例及び参考例においては、RPC を実現する上位プロトコルとして SOAP を採用し、アプリケーションは直接プールを操作して RPC を実現しているが、アプリケーションとプールとの間に CORBA (Common Object Request Broker Architecture) や JAV A (登録商標) RMI (Remote Method Invocation) とのブリッジ (メッセージ変換機能) を備えることによってアプリケーションの開発効率をさらに向上させてもよい。

すなわち、上述した実施例及び参考例における、HTTP クライアント 1 1 と HTTP サーバ 1 2 との間等でのコマンド及びこれに対するコマンド応答のやり取りは、XML で記述された SOAP メッセージにより行うこととしているが、これに限るものでなく、他の形式で記述されていてもよい。

#### 【0 1 7 2】

また、上述した実施例及び参考例において、SOAP 標準のプロトコルだけでなく、SOAP と MIME マルチパートを組み合わせた独自のプロトコルをもこれに加えて採用することにより、HTTP リクエスト、或いは HTTP レスポンスに含まれる SOAP エンベロープを全く独立したものとして扱うこととするが、SOAP の関連仕様として定義された SOAP アタッチメントによって、HTTP レスポンスに含まれる第 1 パートの SOAP エンベロープに、第 2 パート以降の SOAP エンベロープへのリンクを埋め込んでこれらに関連付けて引き渡す構成にしてもよい。SMTP を用いる電子メールの場合にも同様である。

#### 【0 1 7 3】

更に、SOAP 等の上位プロトコルの下位に位置するデータ通信のプロトコルとして、ここでは実施例として HTTP を採用した例について説明したが、この下位プロトコルについても、FTP (File Transfer Protocol) 等の他のプロトコルを採用してもよい。

さらにまた、通信システムの構成についても、以上説明したものに限られることはない。

#### 【0 1 7 4】



また、この発明によるプログラムは、コンピュータを、他の通信装置を通信相手として通信可能な通信装置である、H T T Pクライアント11、H T T Pサーバ12、あるいは通信装置A、Bのような装置として機能させるためのプログラムであり、このようなプログラムをコンピュータに実行させることにより、上述したような効果を得ることができる。

#### 【0175】

このようなプログラムは、はじめからコンピュータに備えるR O MあるいはH D D等の記憶手段に格納しておいてもよいが、記録媒体であるC D-R O Mあるいはフレキシブルディスク、S R A M、E E P R O M、メモリカード等の不揮発性記録媒体（メモリ）に記録して提供することもできる。そのメモリに記録されたプログラムをコンピュータにインストールしてC P Uに実行させるか、C P Uにそのメモリからこのプログラムを読み出して実行させることにより、上述した各手順を実行させることができる。

さらに、ネットワークに接続され、プログラムを記録した記録媒体を備える外部機器あるいはプログラムを記憶手段に記憶した外部機器からダウンロードして実行させることも可能である。

#### 【産業上の利用可能性】

#### 【0176】

以上説明してきたように、この発明の通信サーバ、通信サーバの制御方法、プログラム及び記録媒体を用いれば、通信要求とそれに対する通信応答とを送受信する複数の通信装置が互いに動作要求及び受信した動作要求に対する動作応答を送受信する通信システムを構成する場合において、通信の効率を上げることができる。

従って、この発明を、このような通信システム又はこのような通信システムを構成する通信サーバに適用することにより、通信の負荷が小さい通信システムを構成することができる。

#### 【図面の簡単な説明】

#### 【0177】

【図1】 この発明の通信装置を用いて構成したこの発明の通信システムの構成例を示す図である。

【図2】 図1に示した通信システムにおける動作要求と動作応答の関係を示す図である。

【図3】 通信プロトコルとしてH T T Pを採用する場合の通信システムの構成例を示す図である。

【図4】 図3に示した通信システムにおける動作要求と動作応答の関係を示す図である。

【図5】 図3に示した通信システムにおける通信シーケンスの例を示す図である。

【図6】 その別の例を示す図である。

【図7】 図3に示したH T T Pクライアント及びH T T Pサーバのハードウェア構成例を示す図である。

【図8】 図3に示したH T T Pクライアントの機能のうち、コマンド及びコマンド応答に関する処理を行うための機能の構成を示す機能ブロック図である。

【図9】 図8に示したクライアントコマンドプールに記憶させるクライアントコマンドシートにおけるデータ構造の例を示す図である。

【図10】 図8に示したサーバコマンドプールに記憶させるサーバコマンドシートにおけるデータ構造の例を示す図である。

#### 【0178】

【図11】 図3に示したH T T PクライアントがH T T Pサーバに送信するH T T Pリクエストの例を示す図である。

【図12】 図3に示したH T T PクライアントがH T T Pサーバから受信するH T T Pレスポンスの例を示す図である。

【図13】 クライアントコマンドを記載したパートの例を示す図である。

【図 1 4】 クライアントコマンドに対する応答を記載したパートの例を示す図である。

【図 1 5】 サーバコマンドを記載したパートの例を示す図である。

【図 1 6】 サーバコマンドに対する応答を記載したパートの例を示す図である。

【図 1 7】 図 3 に示した H T T P クライアントにおける、メッセージの収集及び分配処理の基本動作を示すフローチャートである。

【図 1 8】 図 1 7 のステップ S 1 1 乃至 S 1 4 の部分の処理をより詳細に示すフローチャートである。

【図 1 9】 図 1 7 のステップ S 1 5 以降の部分の処理をより詳細に示すフローチャートである。

【図 2 0】 図 3 に示した H T T P クライアントにおける、サーバコマンドの実行に関する処理の一例を示すフローチャートである。

【 0 1 7 9 】

【図 2 1】 その別の例を示すフローチャートである。

【図 2 2】 図 3 に示した H T T P サーバの機能のうち、コマンド及びコマンド応答に関する処理を行うための機能の構成を示す機能ブロック図である。

【図 2 3】 図 2 2 に示したサーバコマンドプールに記憶させるサーバコマンドシートにおけるデータ構造の例を示す図である。

【図 2 4】 図 2 2 に示したクライアントコマンドプールに記憶させるクライアントコマンドシートにおけるデータ構造の例を示す図である。

【図 2 5】 図 3 に示した H T T P サーバにおける、メッセージの収集及び分配処理の基本動作を示すフローチャートである。

【図 2 6】 図 2 5 のステップ S 1 1 1 乃至 S 1 1 5 の部分の処理をより詳細に示すフローチャートである。

【図 2 7】 図 2 5 のステップ S 1 1 6 以降の部分の処理をより詳細に示すフローチャートである。

【図 2 8】 図 3 に示した H T T P サーバにおける、クライアントコマンドの実行に関する処理の一例を示すフローチャートである。

【図 2 9】 その別の例を示すフローチャートである。

【図 3 0】 通信プロトコルとして S M T P を採用する場合の参考例の通信システムの構成例を示す図である。

【 0 1 8 0 】

【図 3 1】 図 3 0 に示した通信システムにおける動作要求と動作応答の関係を示す図である。

【図 3 2】 図 3 0 に示した通信システムにおける通信シーケンスの例を示す図である。

【図 3 3】 図 3 0 に示した通信装置 A の機能のうち、コマンド及びコマンド応答に関する処理を行うための機能の構成を示す、図 8 と対応する機能ブロック図である。

【図 3 4】 図 3 0 に示した通信装置 A が通信装置 B 宛てに送信する電子メールの例を示す図である。

【図 3 5】 図 3 0 に示した通信装置 A における、メッセージ送信時の処理の基本動作を示すフローチャートである。

【図 3 6】 同じく、メッセージ受信時の処理の基本動作のフローチャートを示すフローチャートである。

【図 3 7】 この発明の通信システムの別の構成例を示す図である。

【符号の説明】

【 0 1 8 1 】

1：第 1 の通信装置

2：第 2 の通信装置

1 0：ネットワーク

1 1：H T T P サーバ

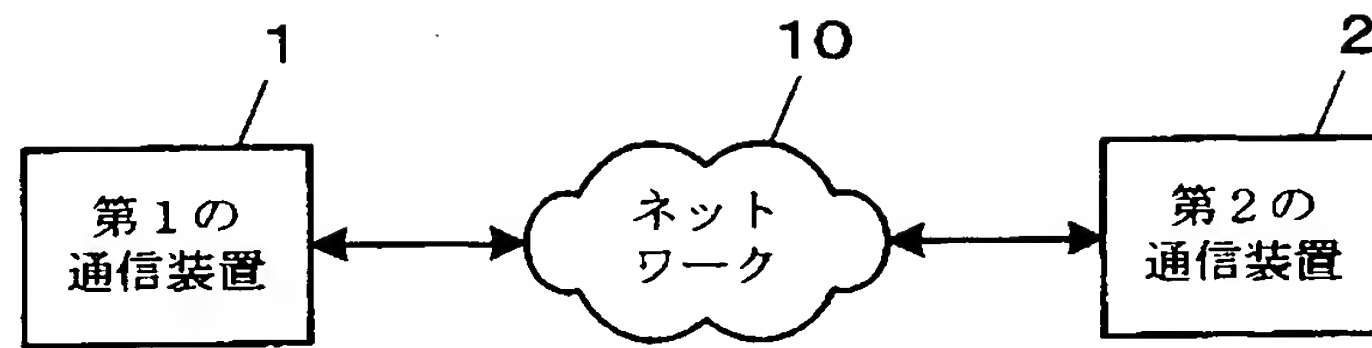
1 2：H T T P クライアント

1 3：インターネット

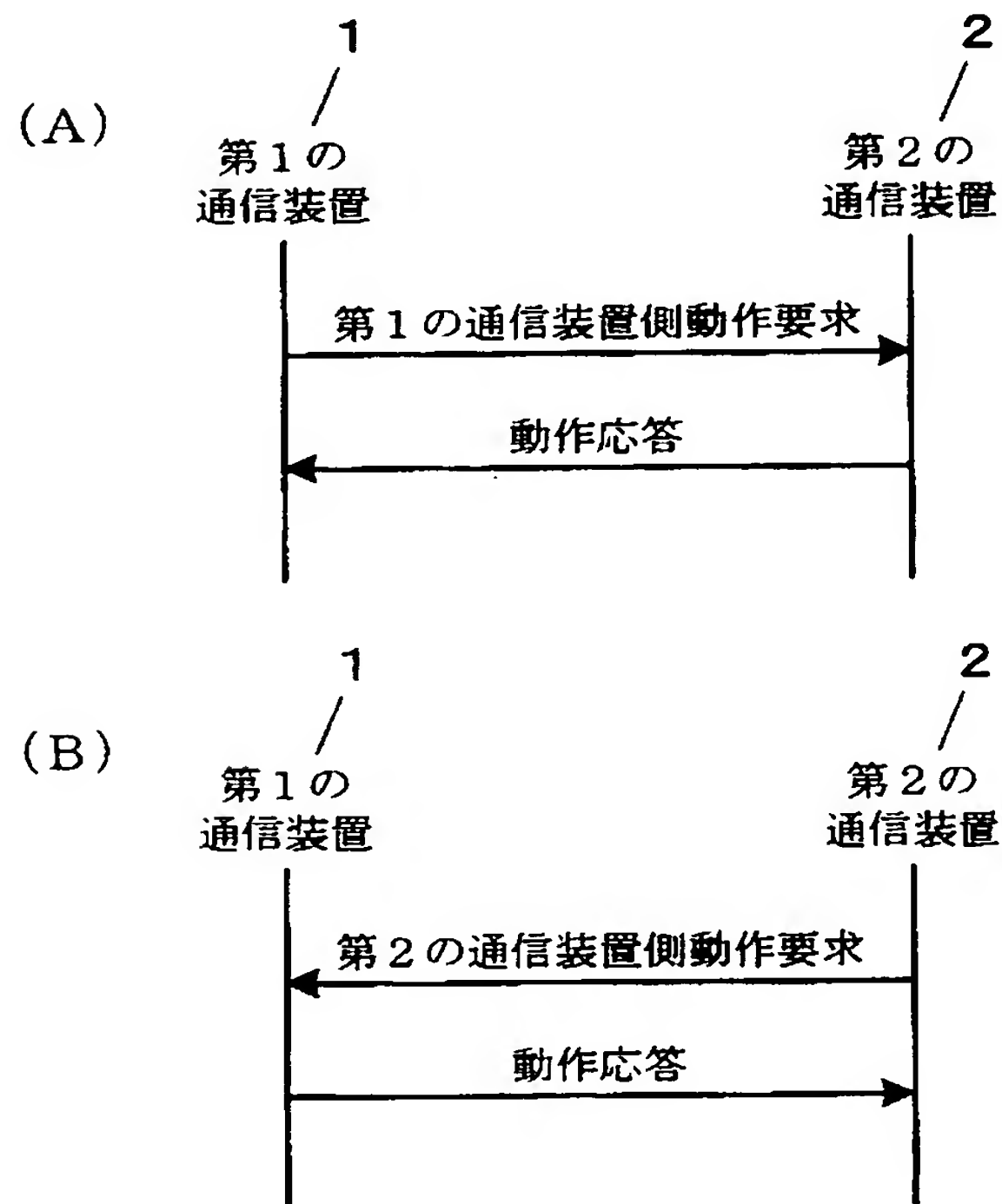
1 4 : ファイアウォール      3 1 : CPU  
3 2 : ROM                      3 3 : RAM  
3 4 : SDカード                3 5 : NIC  
4 1, 1 4 2 : クライアントコマンドプール  
4 2, 1 4 1 : サーバコマンドプール  
4 3 : クライアントコマンド生成手段  
4 4 : サーバコマンド実行結果生成手段  
4 5, 1 4 5 : 送信メッセージ収集手段  
4 6 : HTTPリクエスト送信手段  
4 7 : HTTPレスポンス受信手段  
4 8, 1 4 8 : 受信メッセージ分配手段  
5 1 : 通信装置A側コマンドプール  
5 2 : 通信装置B側コマンドプール  
5 3 : 通信装置A側コマンド生成手段  
5 4 : 通信装置B側コマンド実行結果生成手段  
5 6 : メール送信手段          5 7 : メール受信手段  
1 0 0 : 被管理装置            1 0 1 : 仲介装置  
1 0 2 : 管理装置      1 4 3 : サーバコマンド生成手段  
1 4 4 : クライアントコマンド実行結果生成手段  
1 4 6 : HTTPレスポンス送信手段  
1 4 7 : HTTPリクエスト受信手段



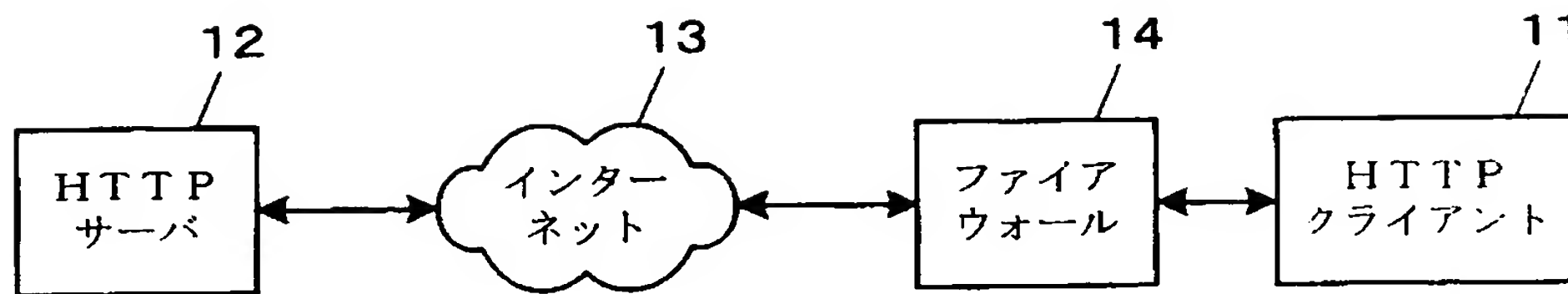
【書類名】 図面  
【図 1】



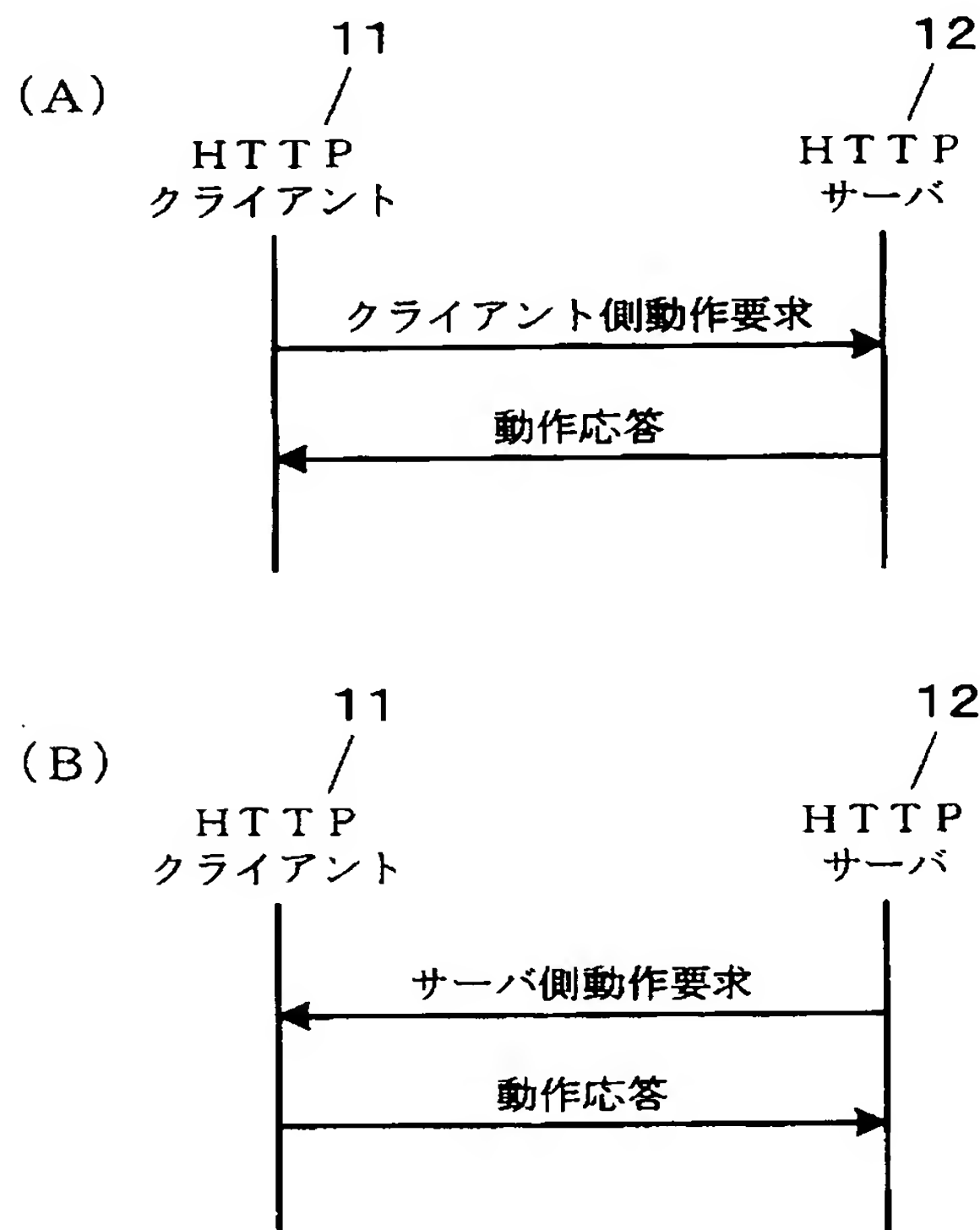
【図 2】



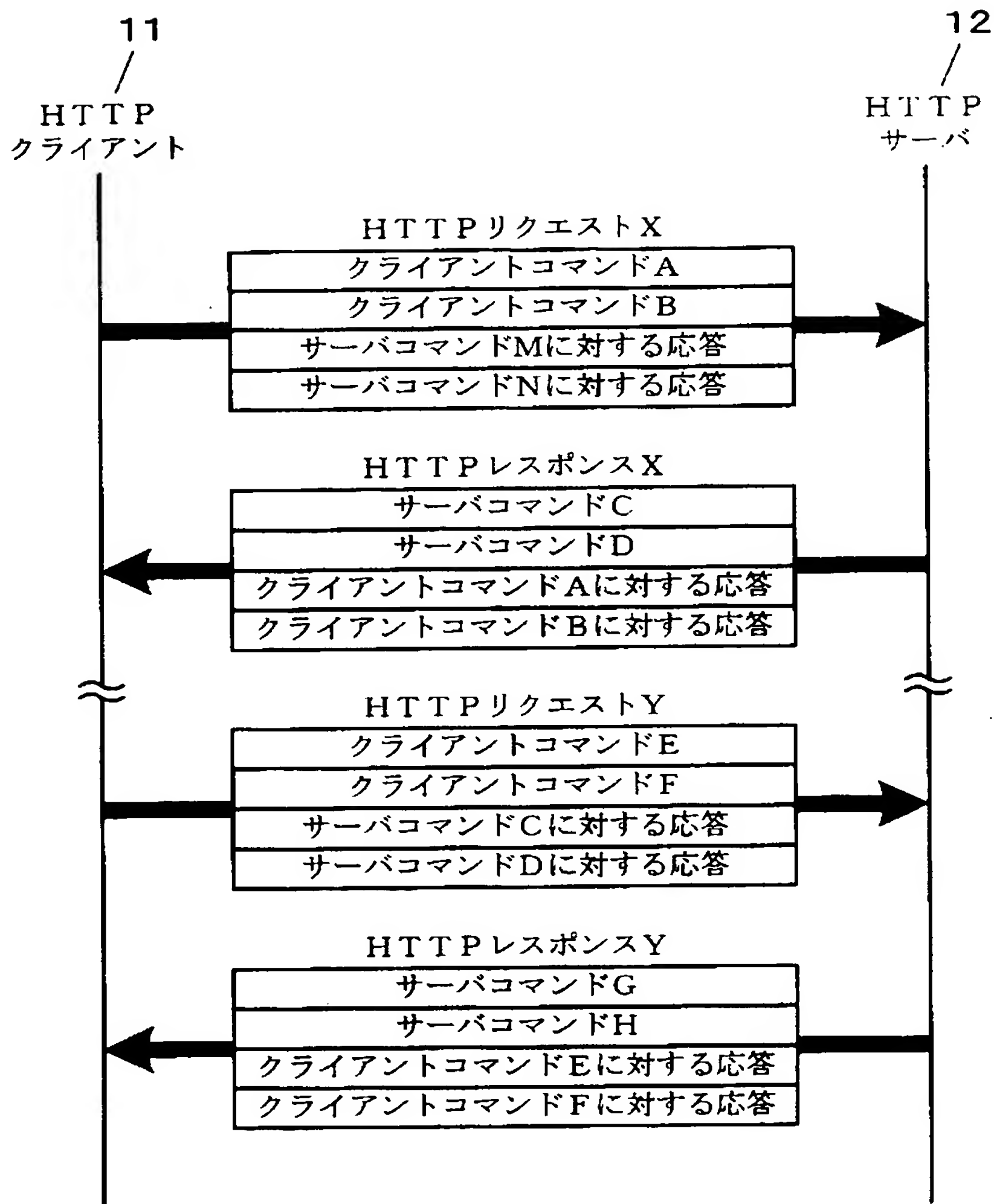
【図 3】



【図 4】

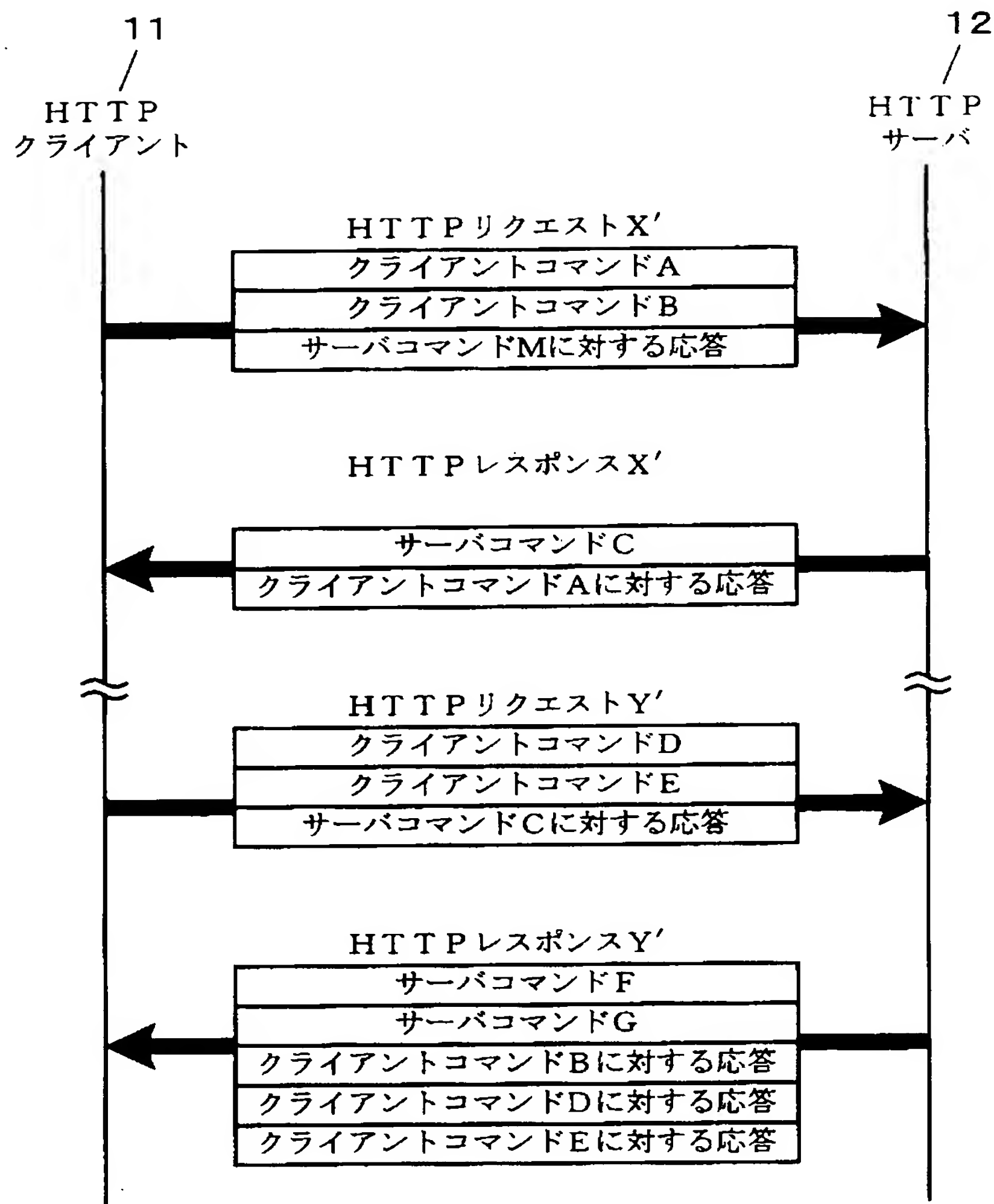


【図 5】

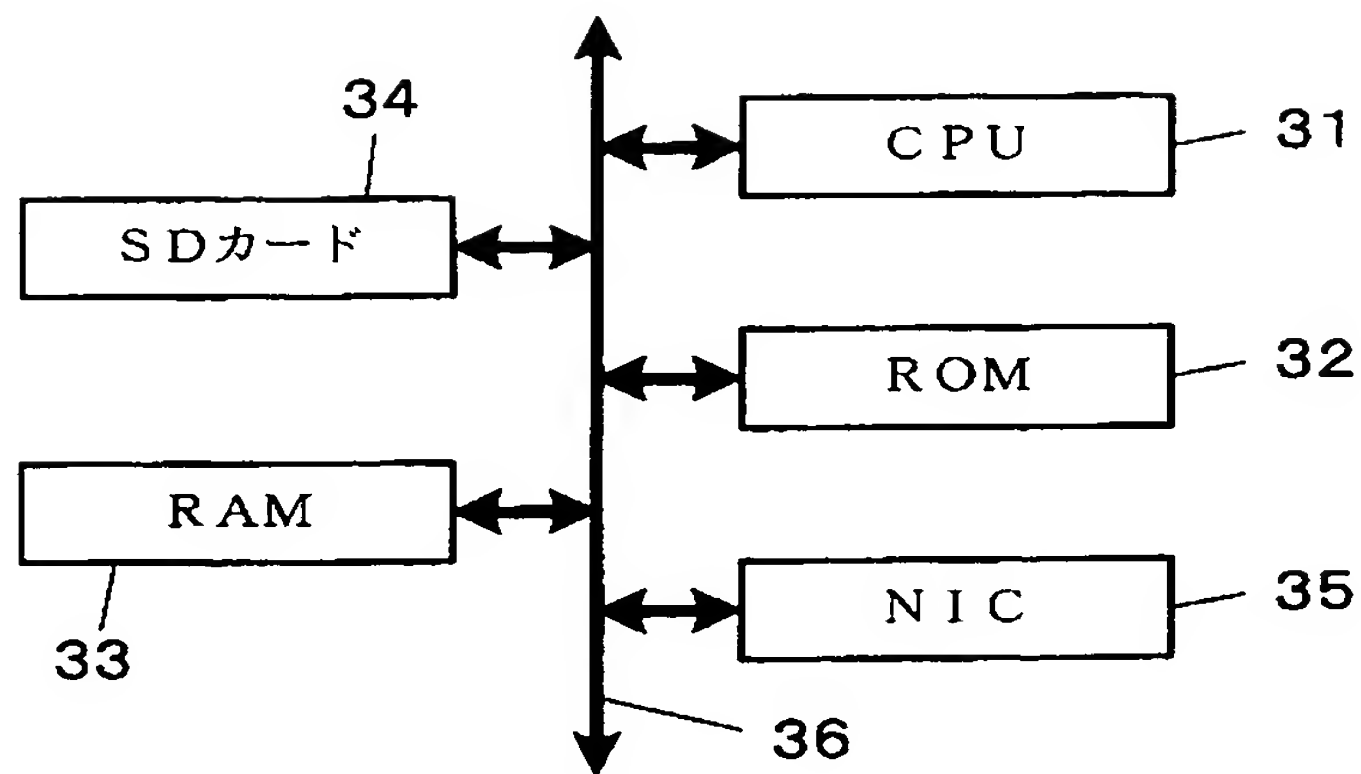




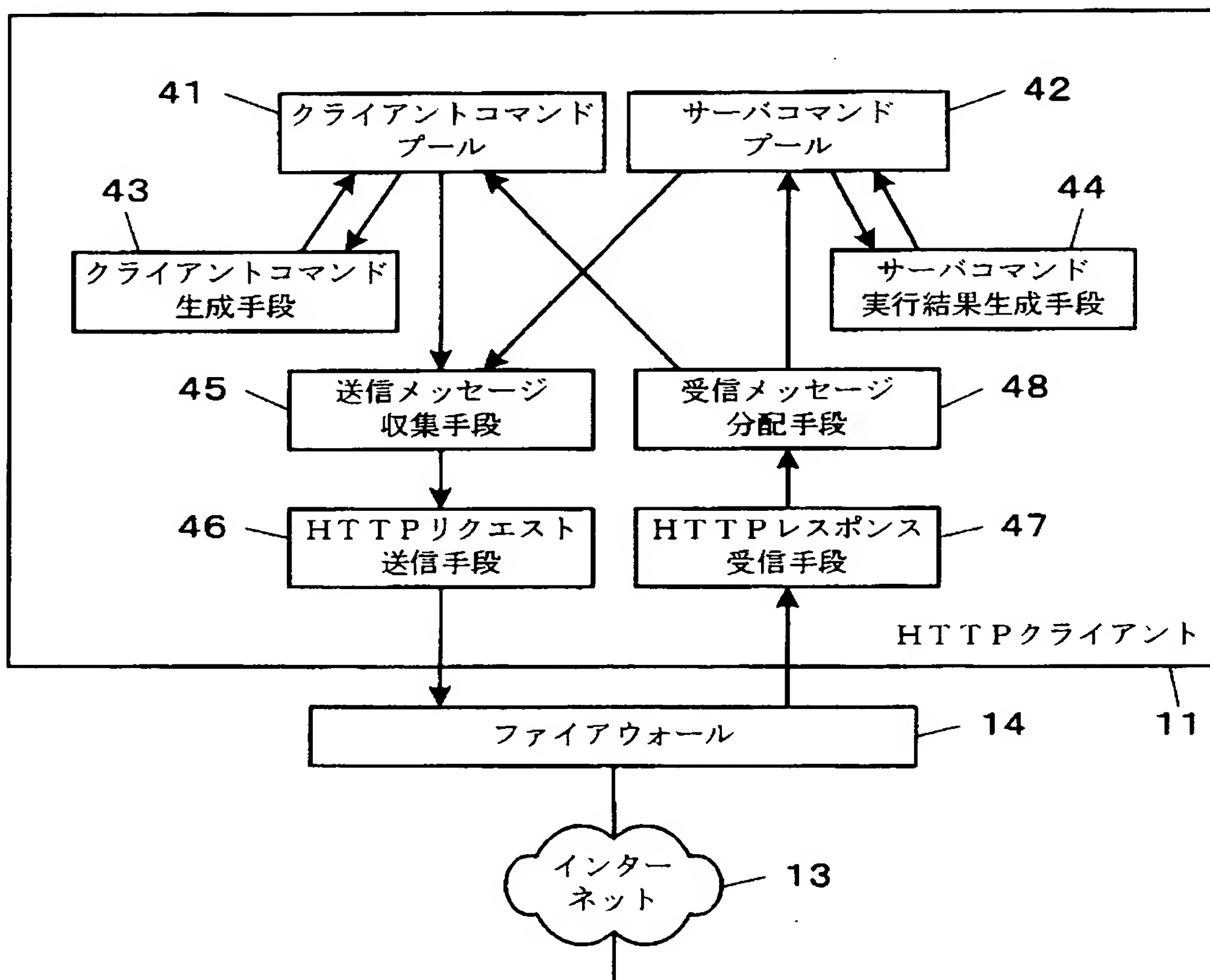
【図 6】



【図 7】



【図 8】



【図 9】

HTTPクライアントにおける  
クライアントコマンドシート

コマンドID
メソッド名（異常通知等）
入力パラメータ（異常内容等）
状態（初期値：未処理）
クライアントコマンド実行結果の通知先
出力パラメータ（応答取得まで空）

【図 1 0】

HTTPクライアントにおける  
サーバコマンドシート

コマンドID
メソッド名（カウンタ取得等）
入力パラメータ
状態（初期値：未処理）
出力パラメータ（処理終了まで空）
サーバコマンドの通知先



【図 11】

## HTTP リクエスト

```
POST /aaa HTTP/1.1
Content-Type:multipart/mixed;boundary=MIME_boundary
Content-Length:nnnn
```

```
--MIME_boundary
Content-Type:text/xml;charset=UTF-8
Content-Transfer-Encoding:8bit
SOAPAction:"SOAP Action URI"
X-SOAP-Type:Request
```

第1パート

```
<s:Envelope>
  <!--SOAP Request-->
</s:Envelope>
```

```
--MIME_boundary
Content-Type:text/xml;charset=UTF-8
Content-Transfer-Encoding:8bit
X-SOAP-Type:Response
```

第2パート

```
<s:Envelope>
  <!--SOAP Response-->
</s:Envelope>
```

```
--MIME_boundary
Content-Type:text/xml;charset=UTF-8
Content-Transfer-Encoding:8bit
X-SOAP-Type:Response
```

第3パート

```
<s:Envelope>
  <!--SOAP Response-->
</s:Envelope>
```

```
--MIME_boundary
Content-Type:text/xml;charset=UTF-8
Content-Transfer-Encoding:8bit
X-SOAP-Type:Response
```

第4パート

```
<s:Envelope>
  <!--SOAP Response-->
</s:Envelope>
```

```
--MIME_boundary--
```

【図 12】

## HTTPレスポンス

```
HTTP/1.1 200 OK
Content-Type:multipart/mixed;boundary=MIME_boundary
Content-Length:nnnn

--MIME_boundary
Content-Type:text/xml;charset=UTF-8
Content-Transfer-Encoding:8bit
SOAPAction:"SOAP Action URI"
X-SOAP-Type:Request
<s:Envelope>
  <!--SOAP Request-->
</s:Envelope>
--MIME_boundary
Content-Type:text/xml;charset=UTF-8
Content-Transfer-Encoding:8bit
X-SOAP-Type:Response
<s:Envelope>
  <!--SOAP Response-->
</s:Envelope>
--MIME_boundary
Content-Type:text/xml;charset=UTF-8
Content-Transfer-Encoding:8bit
X-SOAP-Type:Response
<s:Envelope>
  <!--SOAP Response-->
</s:Envelope>
--MIME_boundary
Content-Type:text/xml;charset=UTF-8
Content-Transfer-Encoding:8bit
X-SOAP-Type:Response
<s:Envelope>
  <!--SOAP Response-->
</s:Envelope>
--MIME_boundary--
```

第1パート

第2パート

第3パート

第4パート

## 【図 1 3】

## クライアントコマンドのパート例

```
Content-Type: text/xml; charset=UTF-8
Content-Transfer-Encoding: 8bit
X-SOAP-Type: Request
SOAPAction: "http://www.foo.com/server/errorNotification"

<s:Envelope
  xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:se="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:n="http://www.foo.com/header"
  xmlns:ns="http://www.foo.com/server"
  s:encodingStyle="http://schemas.xmlsoap.org/soap/encoding">

  <s:Header>
    <n:要求ID>12345</n:要求ID>
  </s:Header>
  </s:Body>
    <ns:異常通知>
      <エラーID>1111</エラーID>
      <説明>ハードディスクドライブ異常</説明>
    </ns:異常通知>
  </s:Body>
</s:Envelope>
```

## 【図 1 4】

## クライアントコマンドに対する応答のパート例

```
Content-Type: text/xml; charset=UTF-8
Content-Transfer-Encoding: 8bit
X-SOAP-Type: Response

<s:Envelope
  xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:se="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:n="http://www.foo.com/header"
  xmlns:ns="http://www.foo.com/server"
  s:encodingStyle="http://schemas.xmlsoap.org/soap/encoding">

  <s:Header>
    <n:要求ID>12345</n:要求ID>
  </s:Header>
  </s:Body>
    <ns:異常通知Response>
      <受信結果>OK</受信結果>
    </ns:異常通知Response>
  </s:Body>
</s:Envelope>
```

【図 1 5】

サーバコマンドのパート例

```
Content-Type: text/xml; charset=UTF-8
Content-Transfer-Encoding: 8bit
X-SOAP-Type: Request
SOAPAction: "http://www.foo.com/client/getTemperature"

<s:Envelope
  xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:se="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:n="http://www.foo.com/header"
  xmlns:nc="http://www.foo.com/client"
  s:encodingStyle="http://schemas.xmlsoap.org/soap/encoding">

  <s:Header>
    <n:要求ID>98765</n:要求ID>
  </s:Header>
  </s:Body>
    <nc:温度センサ値取得>
      <センサID>3</センサID>
    </nc:温度センサ値取得>
  </s:Body>
</s:Envelope>
```

【図 1 6】

サーバコマンドに対する応答のパート例

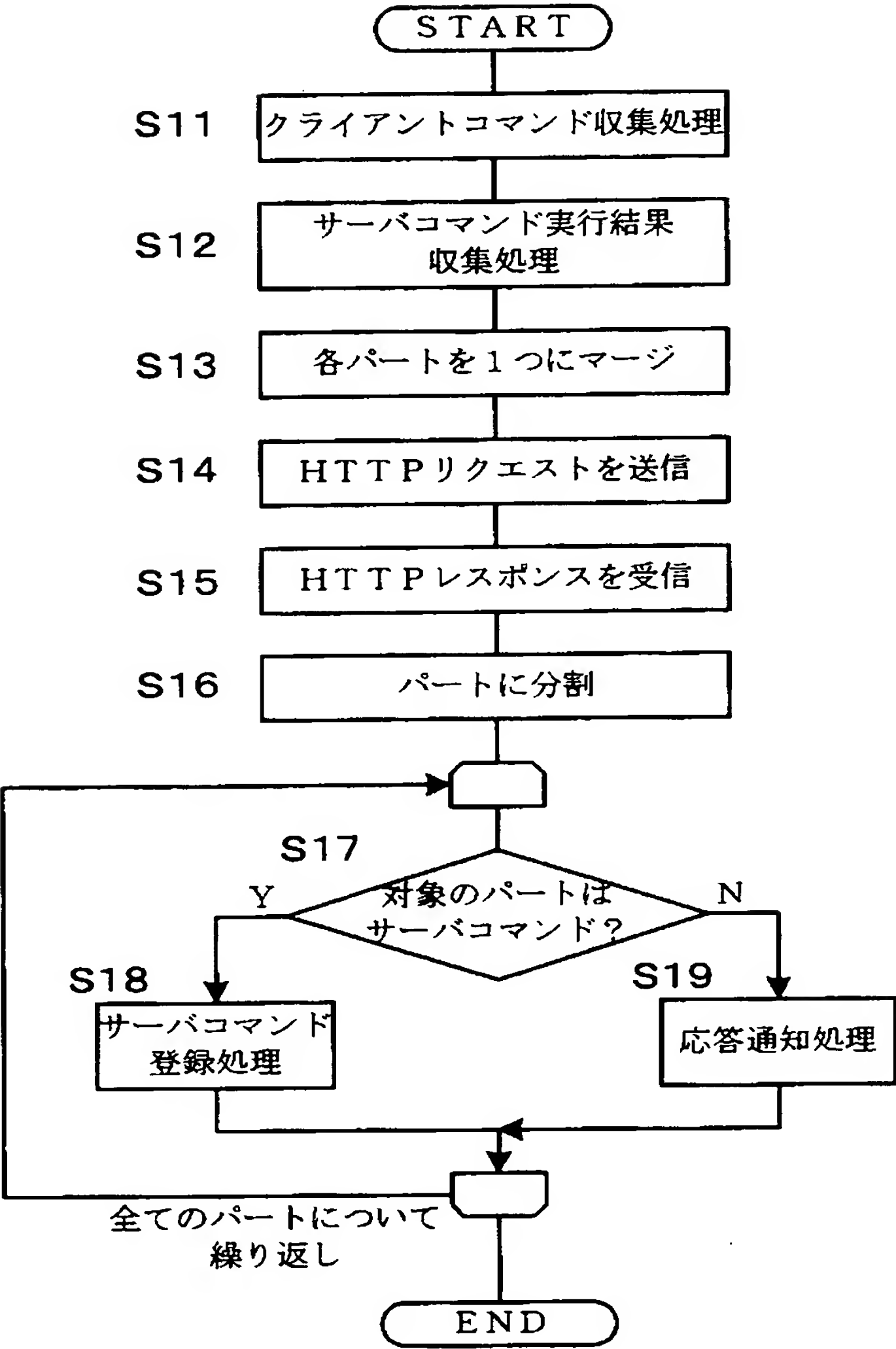
```
Content-Type: text/xml; charset=UTF-8
Content-Transfer-Encoding: 8bit
X-SOAP-Type: Response

<s:Envelope
  xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:se="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:n="http://www.foo.com/header"
  xmlns:nc="http://www.foo.com/client"
  s:encodingStyle="http://schemas.xmlsoap.org/soap/encoding">

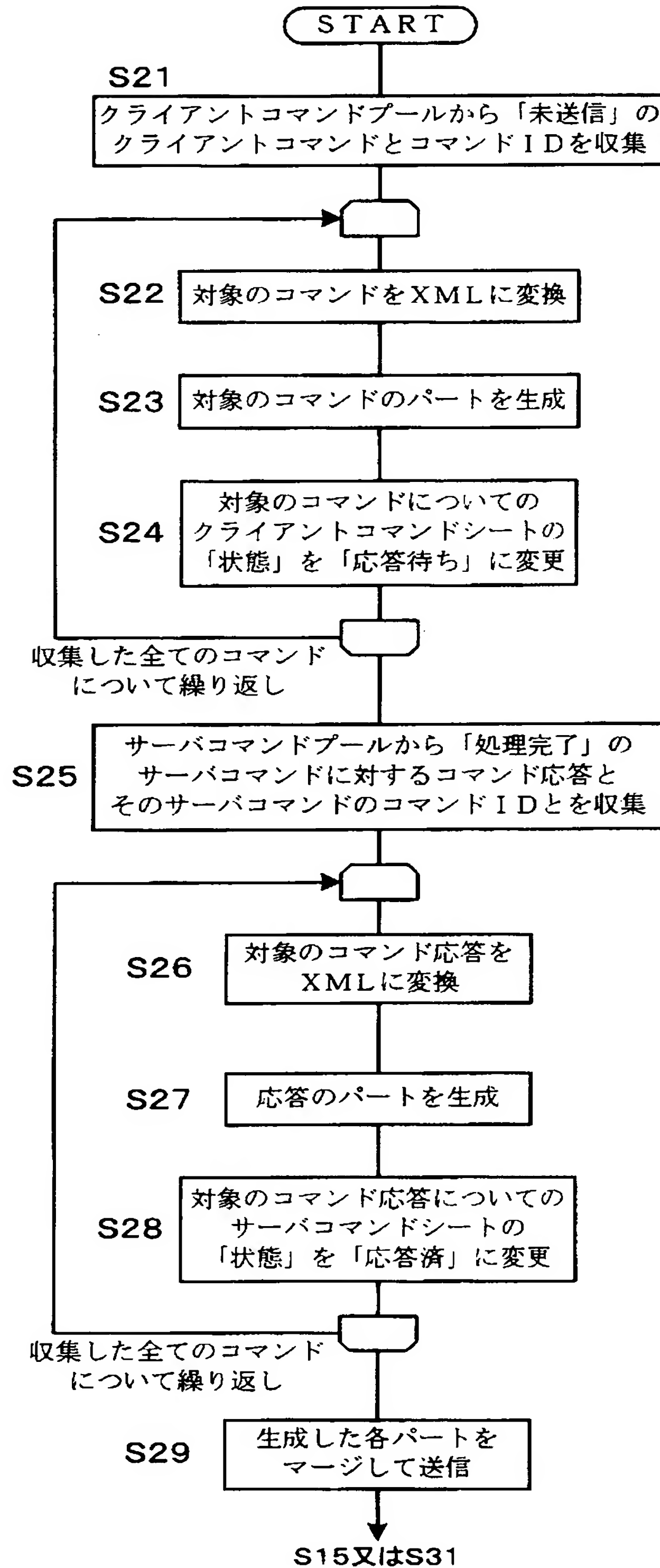
  <s:Header>
    <n:要求ID>98765</n:要求ID>
  </s:Header>
  </s:Body>
    <nc:温度センサ値取得Response>
      <温度>52</温度>
    </nc:温度センサ値取得Response>
  </s:Body>
</s:Envelope>
```



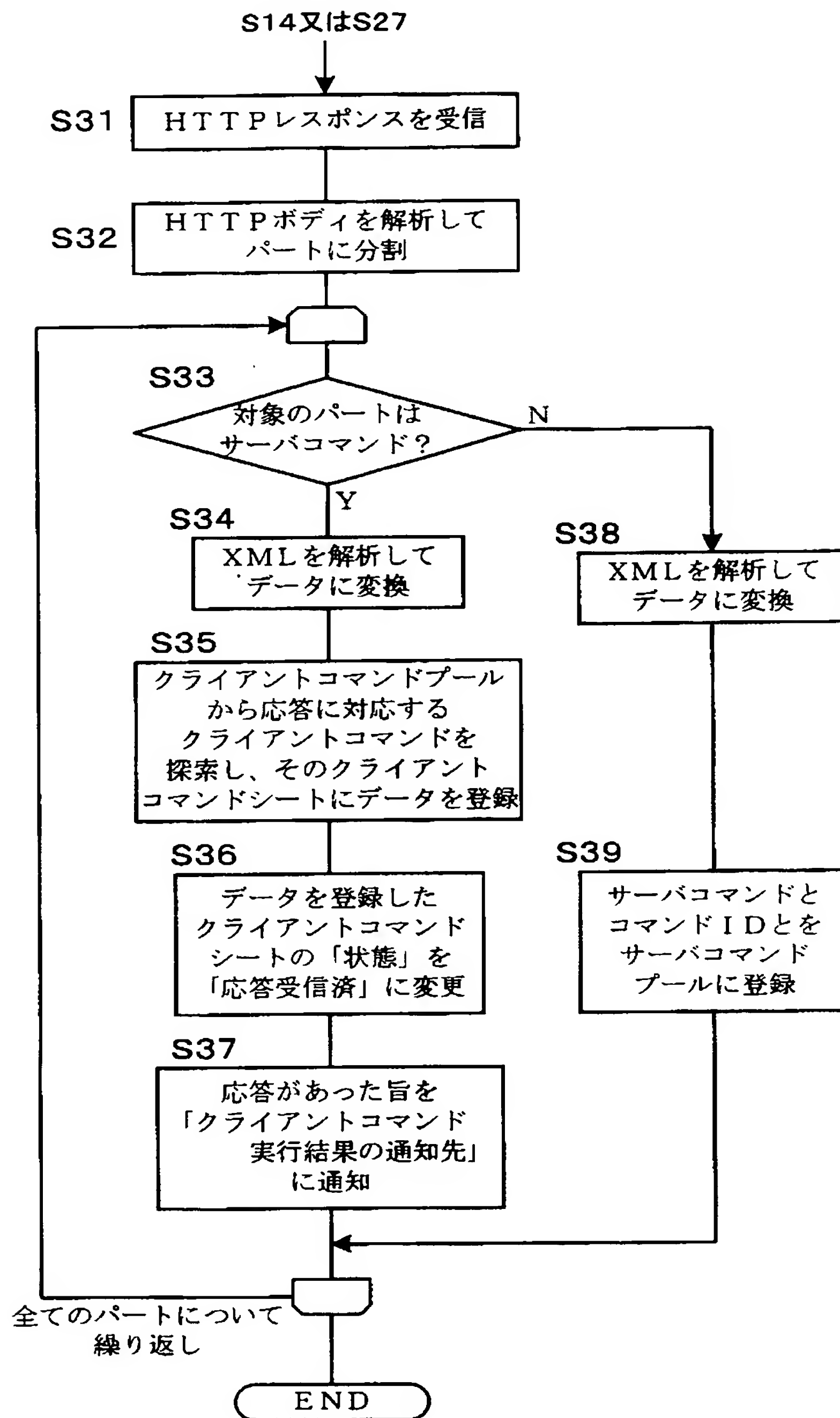
【図 17】



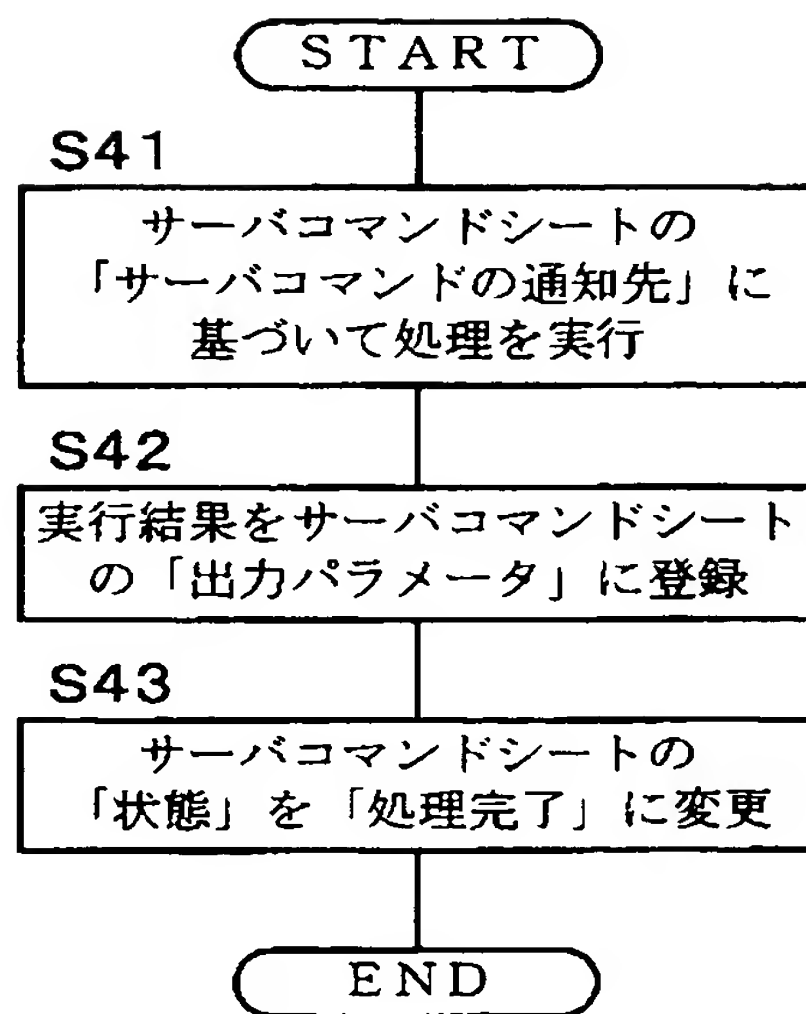
【図 18】



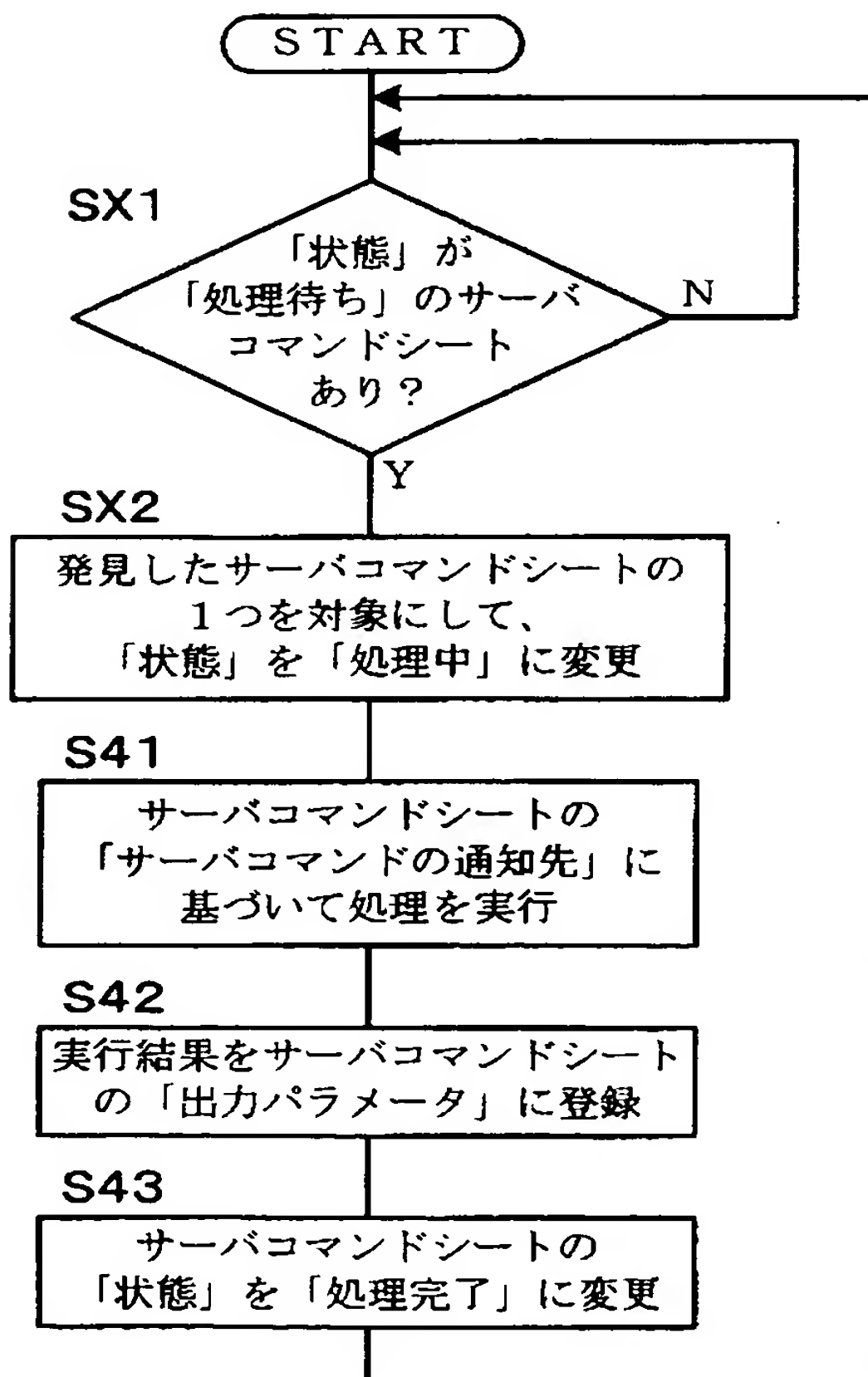
【図 19】



【図 20】

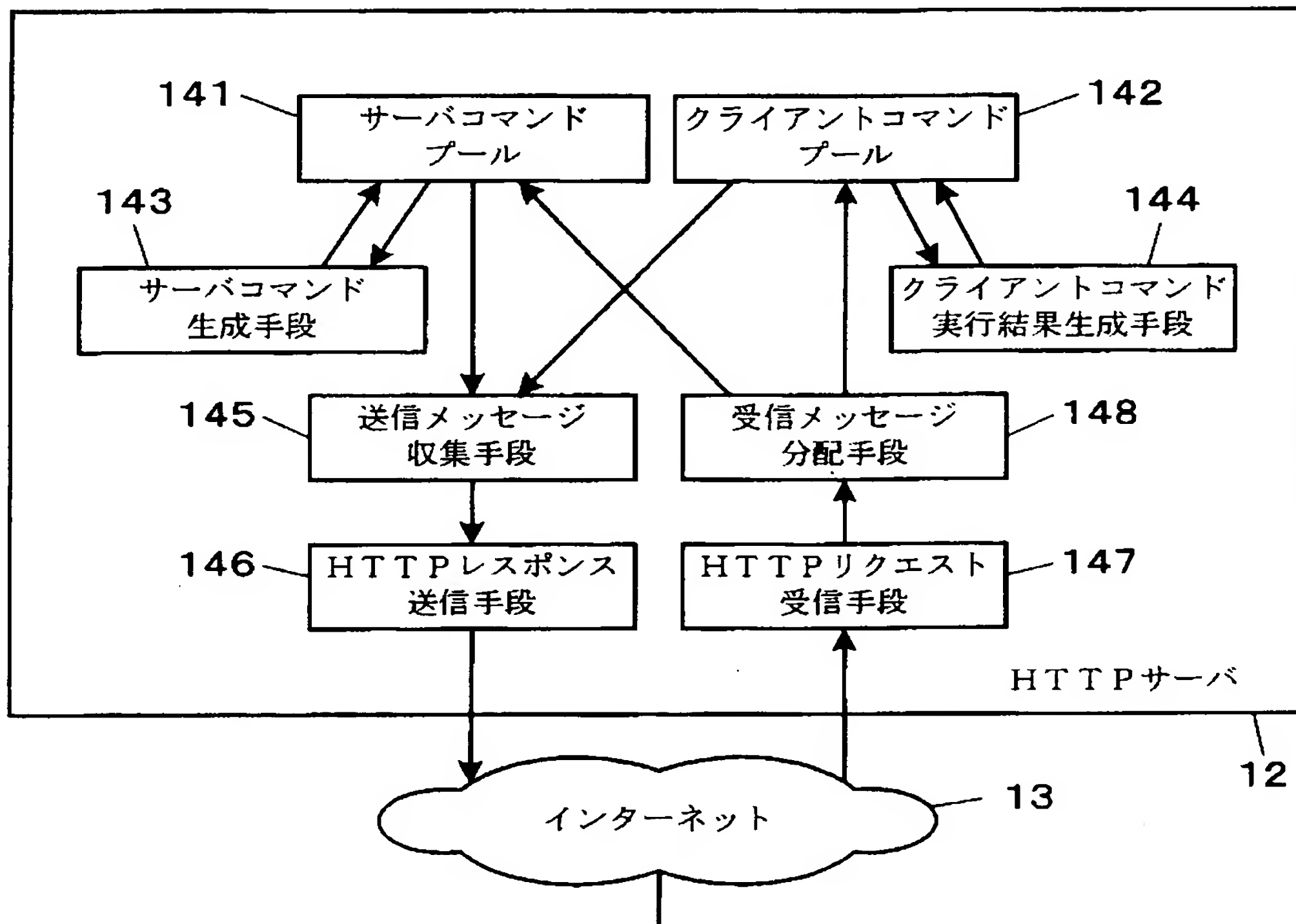


【図 21】





【図 22】



【図 2 3】

HTTPサーバにおける  
サーバコマンドシート

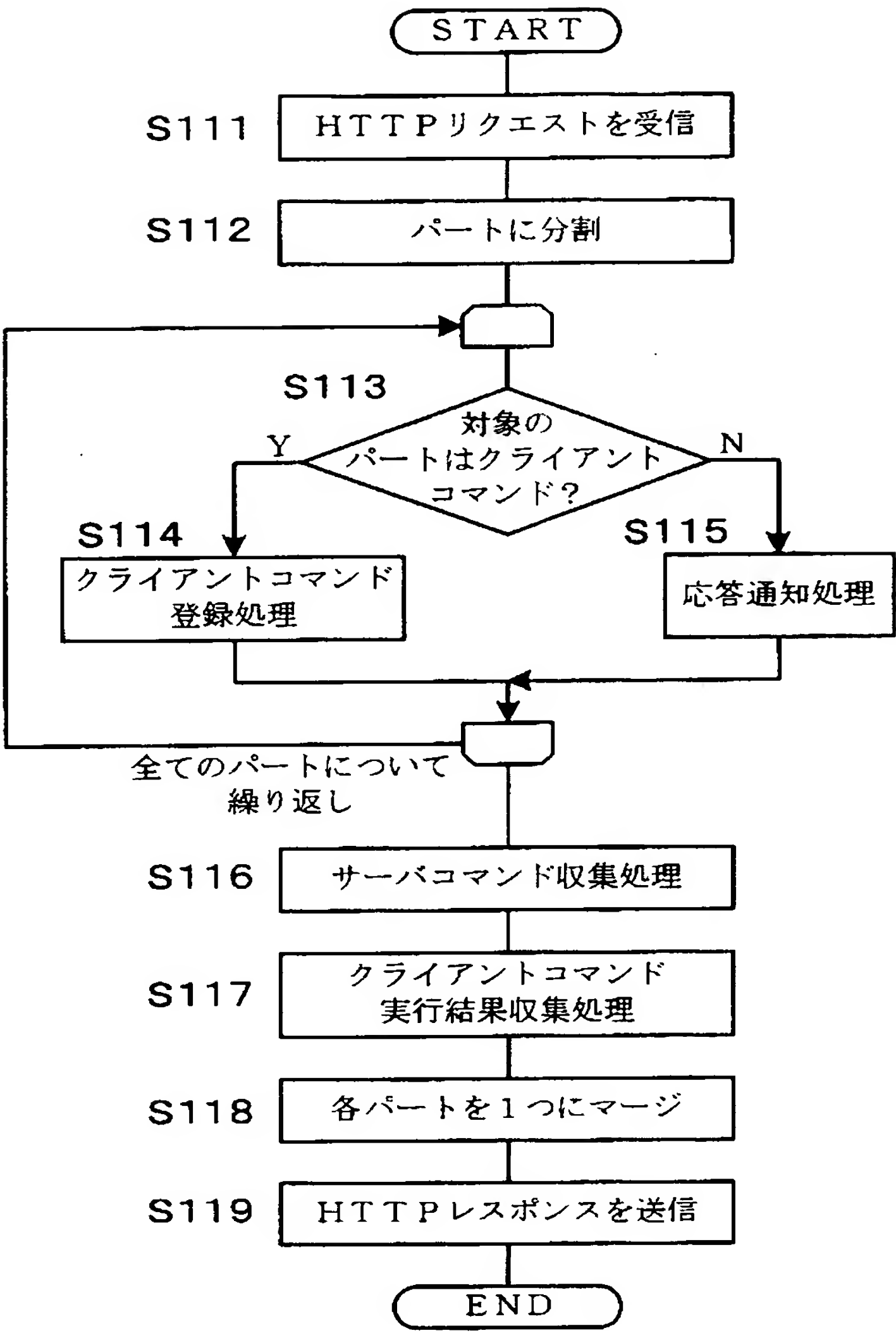
コマンドID  
メソッド名（カウンタ取得等）  
入力パラメータ  
状態（初期値：未処理）  
サーバコマンド実行結果の通知先  
出力パラメータ（応答取得まで空）

【図 2 4】

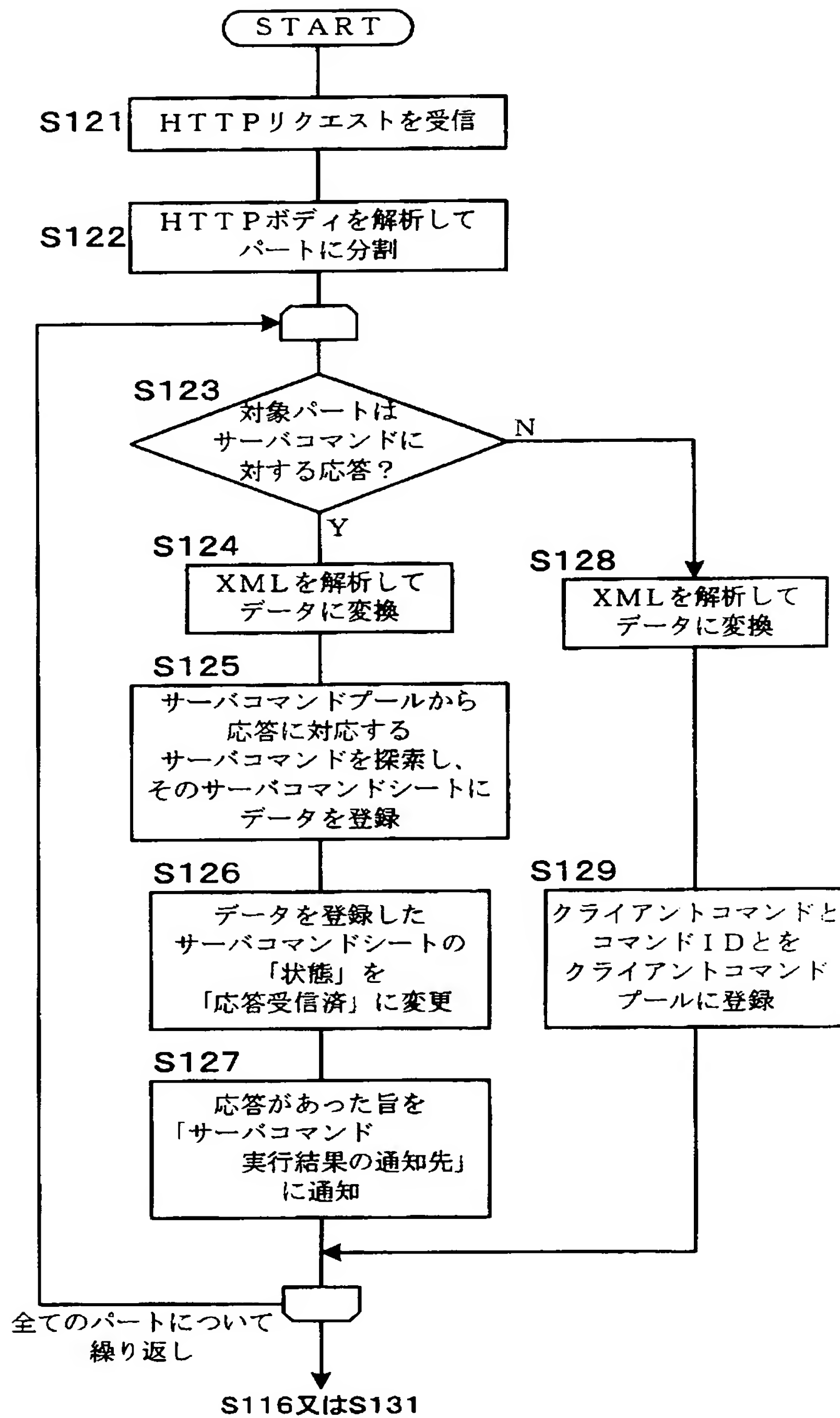
HTTPサーバにおける  
クライアントコマンドシート

コマンドID  
メソッド名（異常通知等）  
入力パラメータ（異常内容等）  
状態（初期値：未処理）  
出力パラメータ（処理終了まで空）  
クライアントコマンドの通知先

【図 2 5】

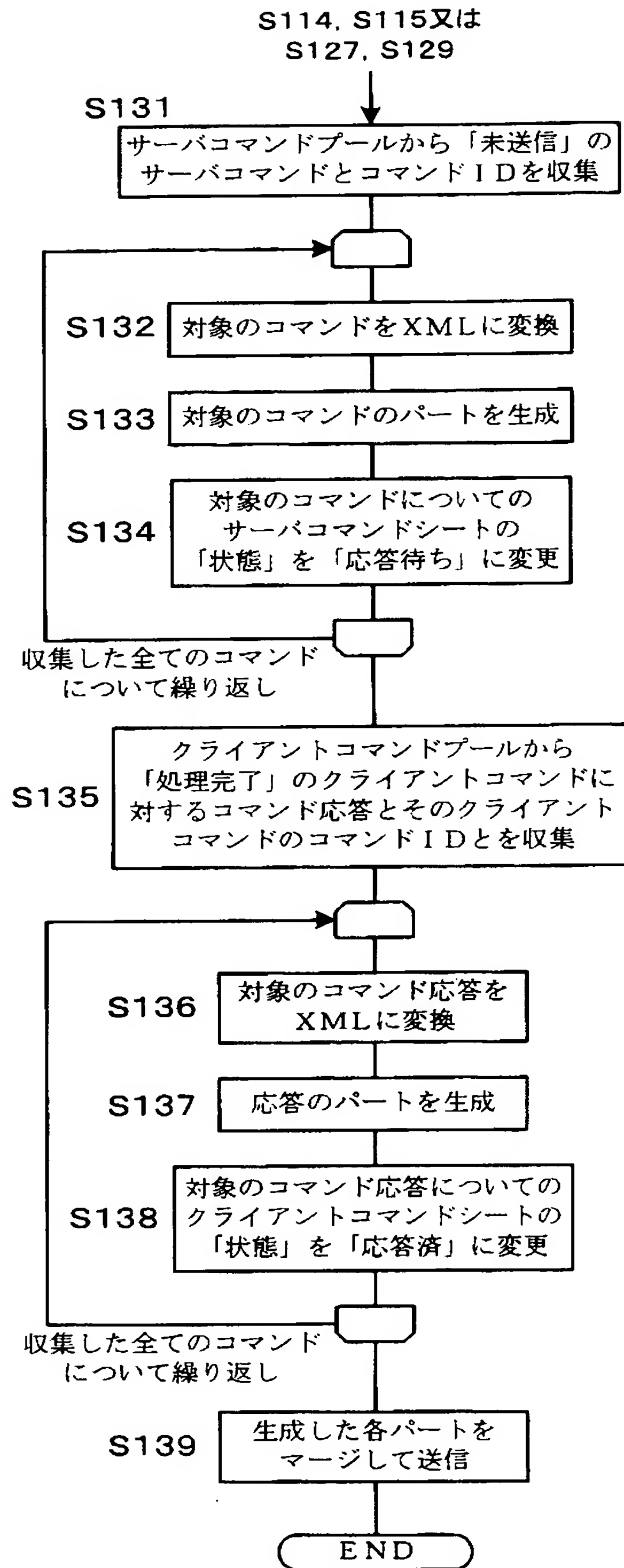


【図 26】

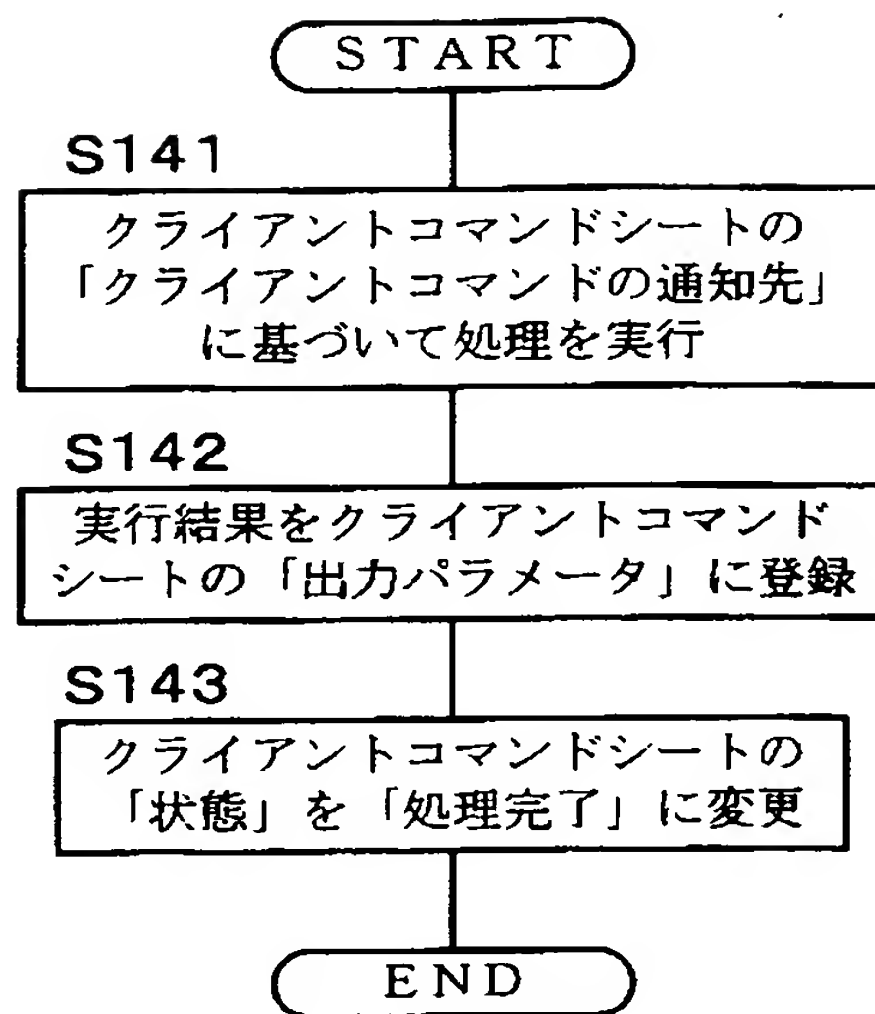




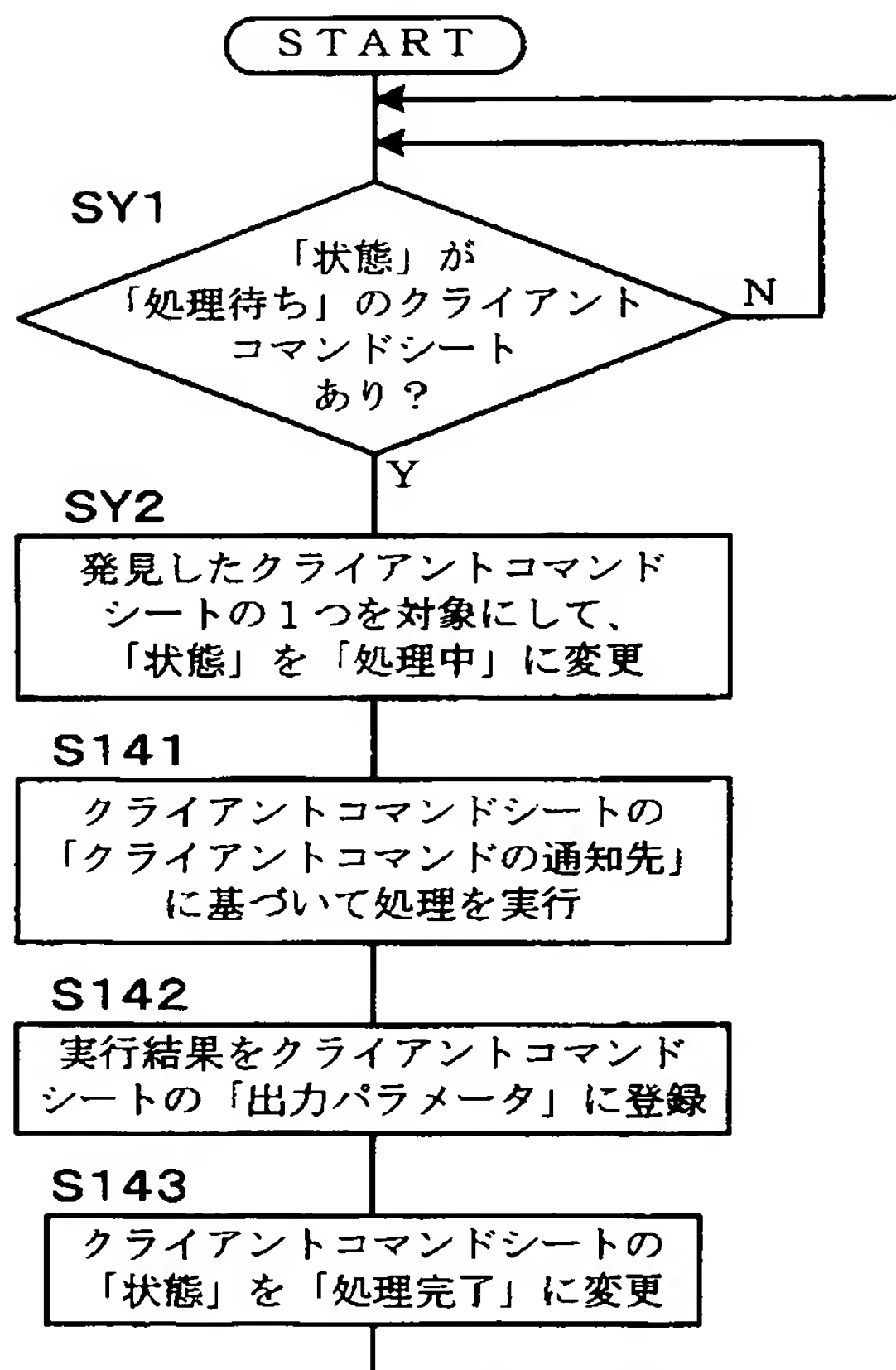
【図 27】



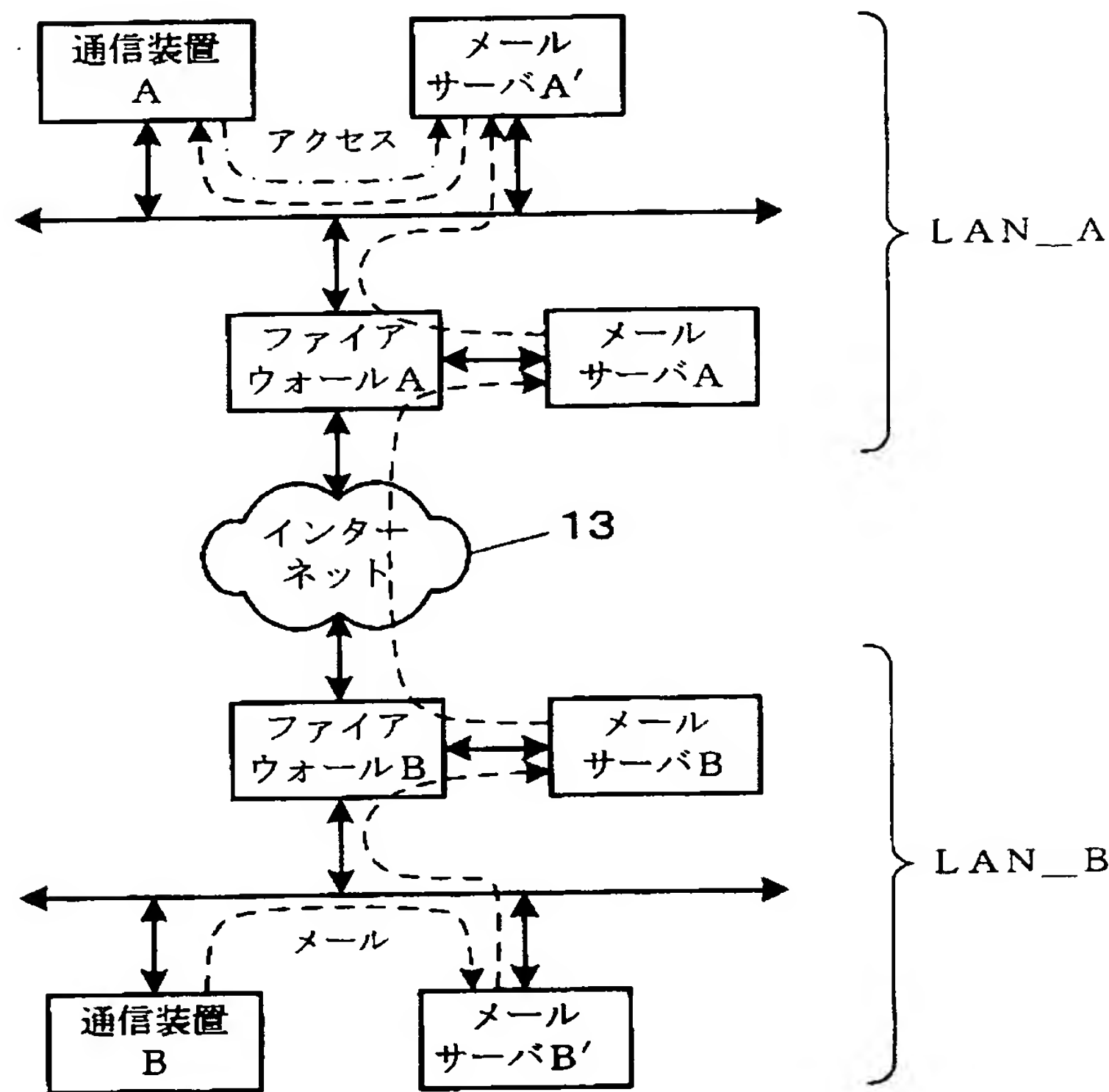
【図 28】



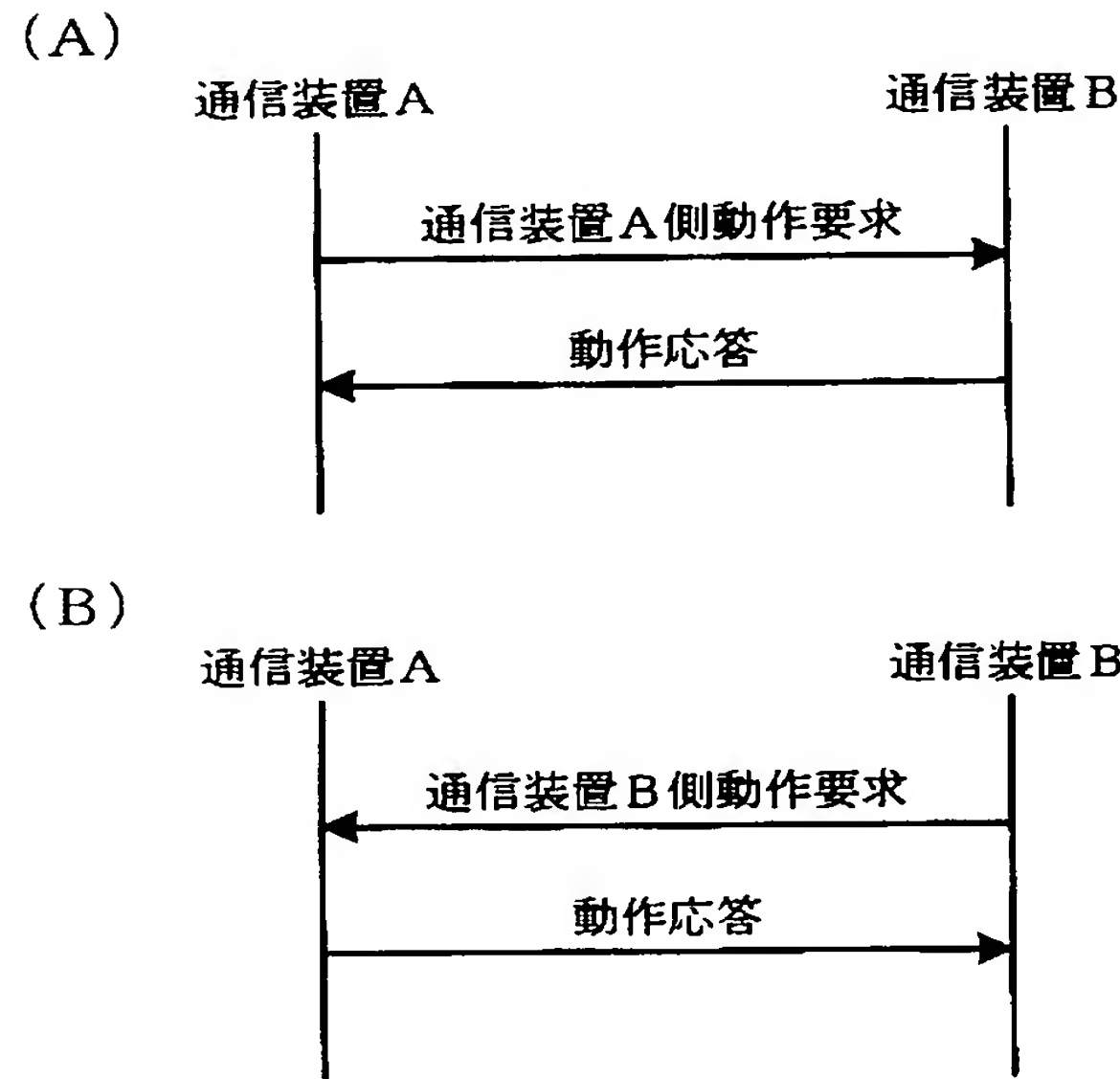
【図 29】



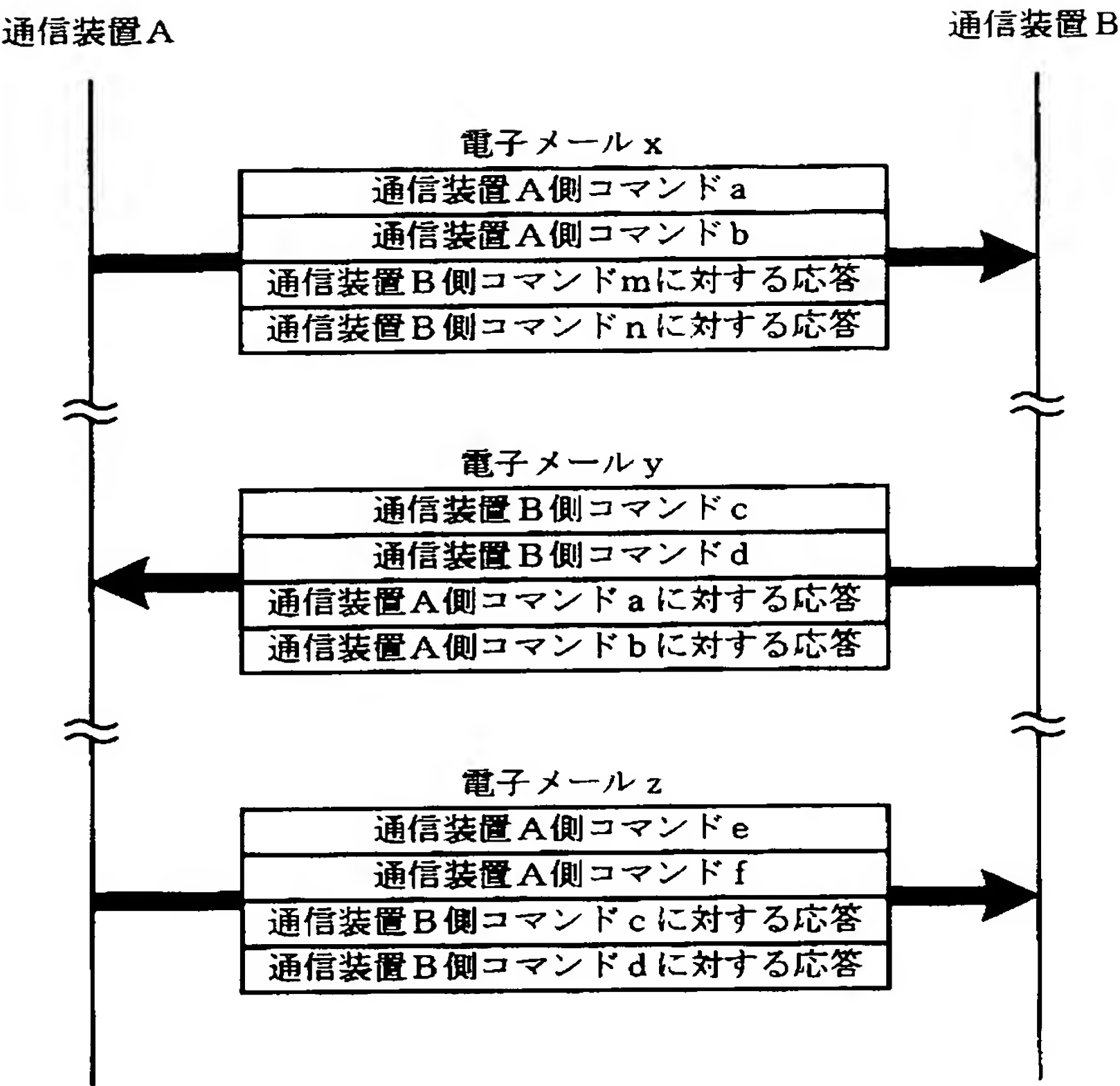
【図 3 0】



【図 3 1】

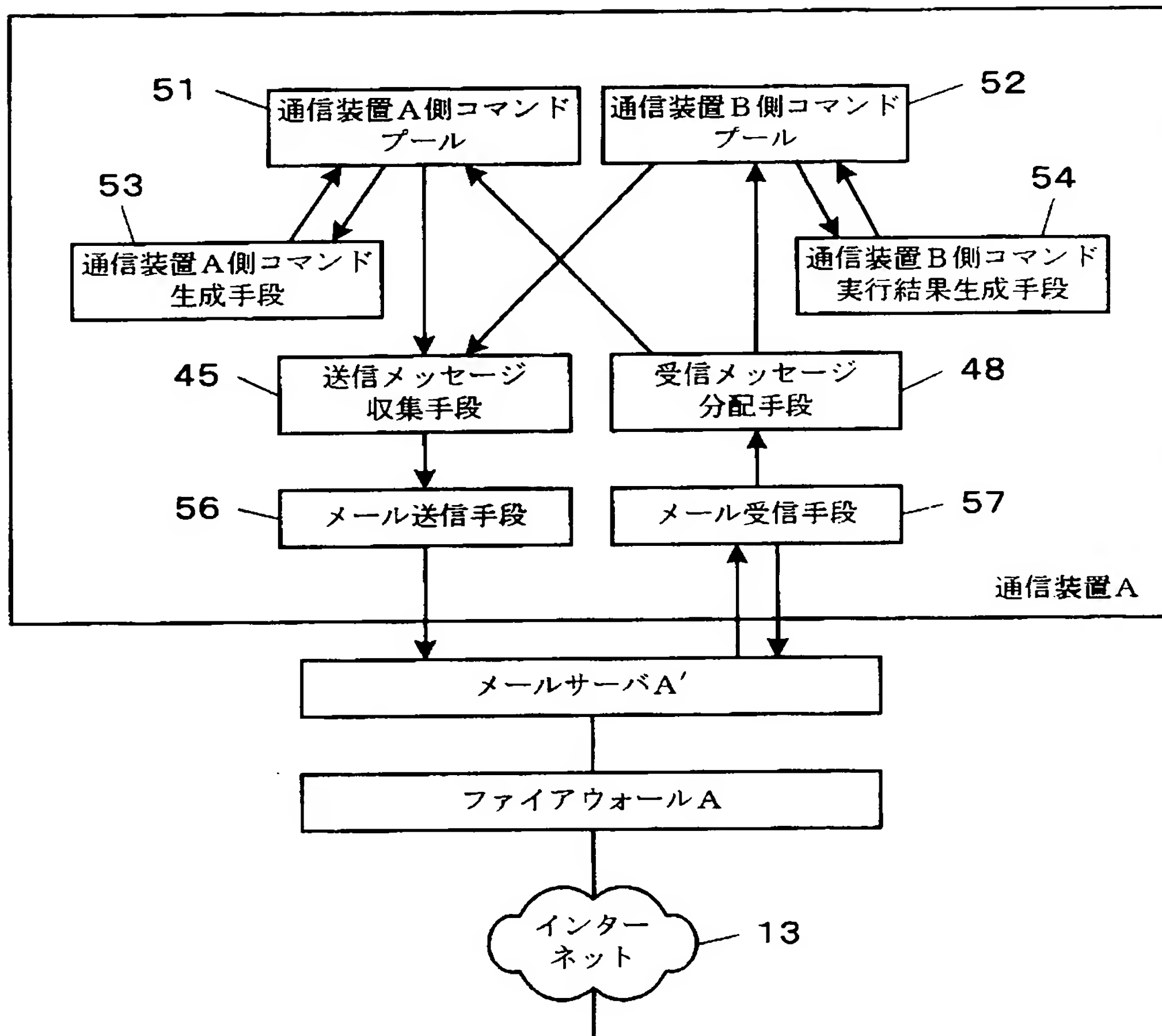


【図 3 2】





【図 33】



【図 3 4】

```
From: deviceA@foo.com
To: deviceB@bar.com
Subject: Multi Message #00000001
Date: Wed, 30 Jul 2003 10:00:00 +0900
Content-Type: multipart/mixed;boundary=MIME_boundary
Content-Length:nnnn

--MIME_boundary
Content-Type:text/xml;charset=iso-2022-jp
Content-Transfer-Encoding:7bit
SOAPAction:"SOAP Action URI"
X-SOAP-Type:Request
<s:Envelope>
  <--SOAP Request-->
</s:Envelope>

--MIME_boundary
Content-Type:text/xml;charset=iso-2022-jp
Content-Transfer-Encoding:7bit
X-SOAP-Type:Response
<s:Envelope>
  <--SOAP Response-->
</s:Envelope>

--MIME_boundary
Content-Type:text/xml;charset=iso-2022-jp
Content-Transfer-Encoding:7bit
X-SOAP-Type:Response
<s:Envelope>
  <--SOAP Response-->
</s:Envelope>

--MIME_boundary
Content-Type:text/xml;charset=iso-2022-jp
Content-Transfer-Encoding:7bit
X-SOAP-Type:Response
<s:Envelope>
  <--SOAP Response-->
</s:Envelope>

--MIME_boundary--
```

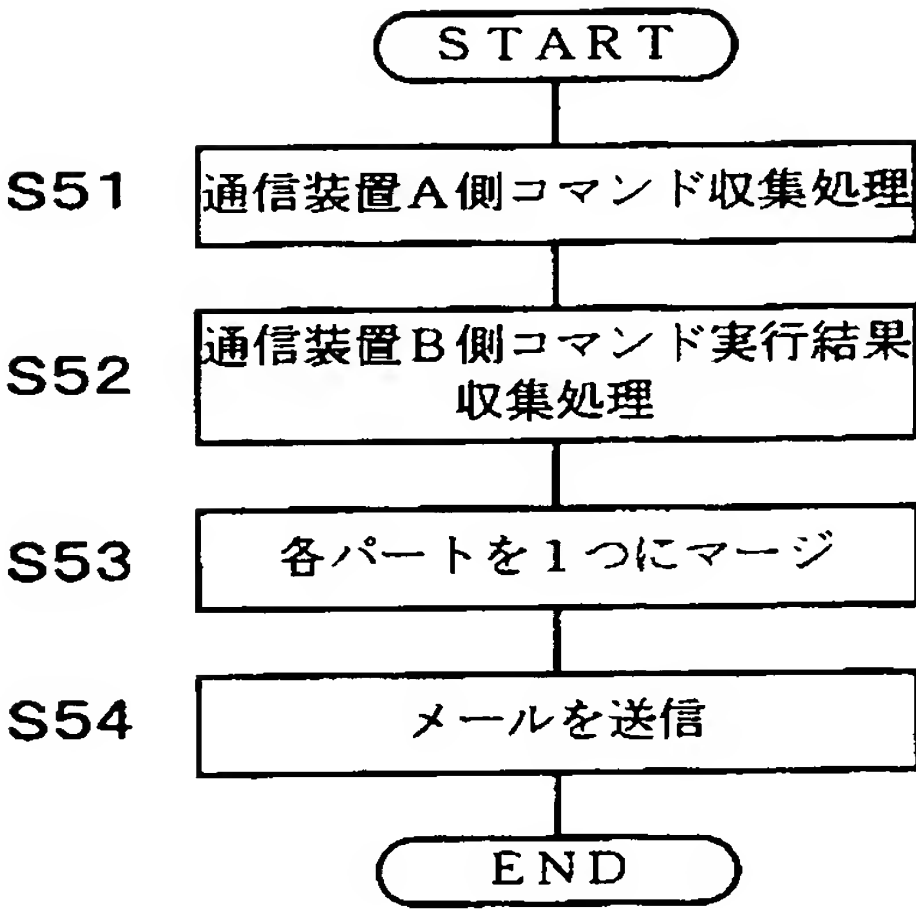
第 1 パート

第 2 パート

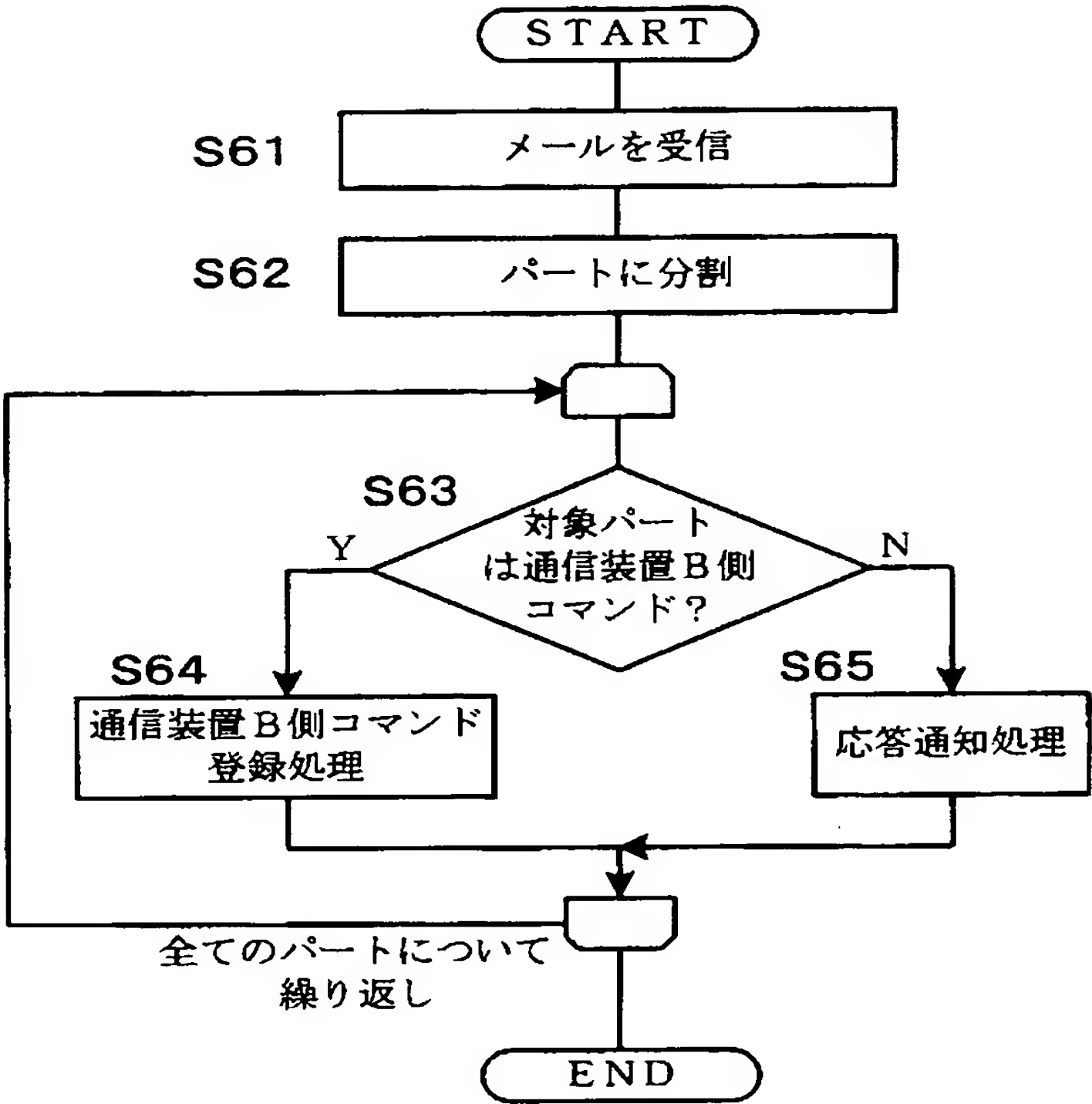
第 3 パート

第 4 パート

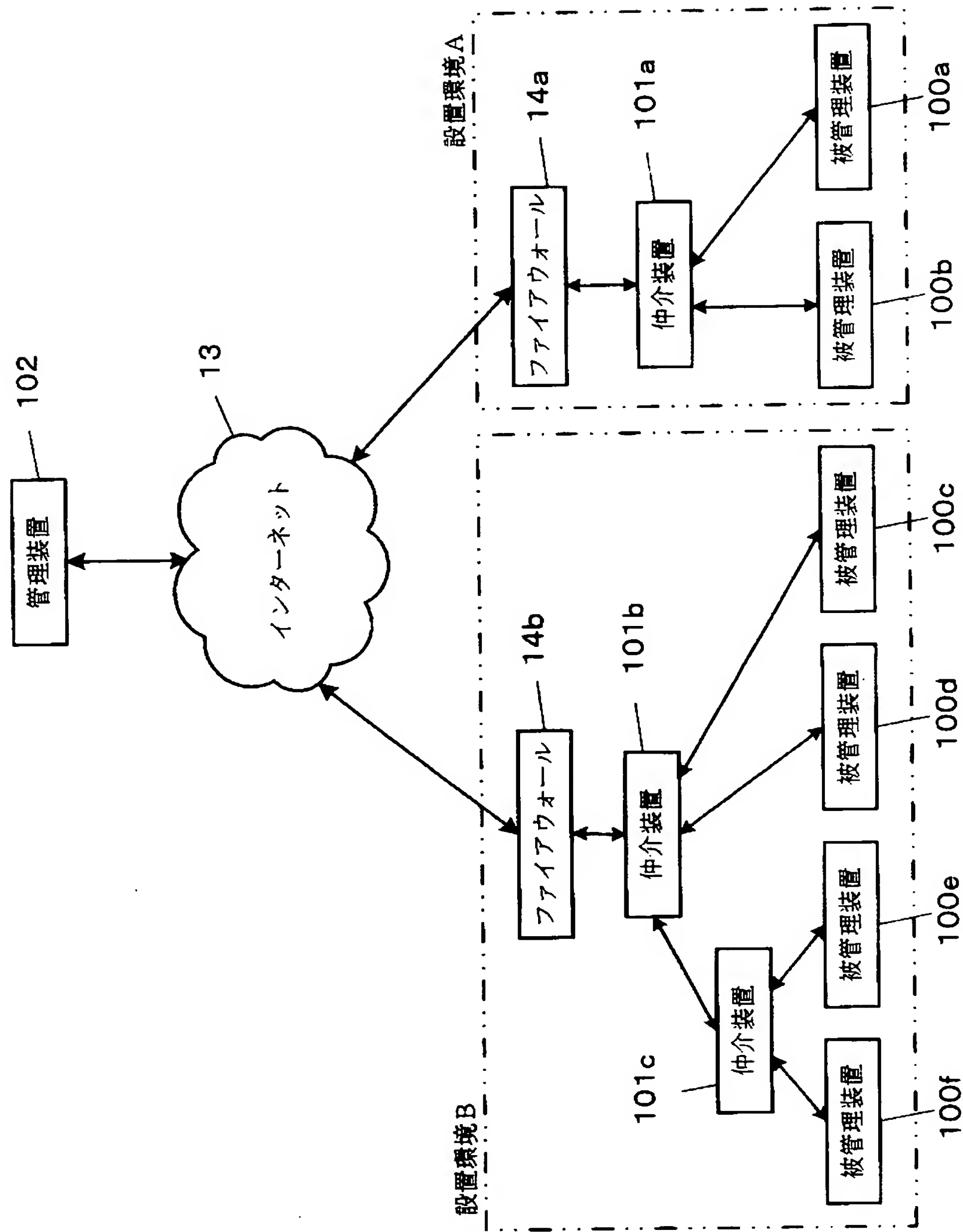
【図 3 5】



【図 3 6】



【図 37】



**【書類名】 要約書****【要約】**

**【課題】** 通信要求とそれに対する通信応答とを送受信する複数の通信装置が互いに動作要求及び受信した動作要求に対する動作応答を送受信する場合において、通信の効率を上げる。

**【解決手段】** H T T Pサーバ 1 2 において、H T T Pクライアントに送信した動作要求に対する動作応答とH T T Pクライアントからの動作要求とをH T T Pリクエスト受信手段 1 4 7 によって一括して受信し、受信メッセージ分配手段 1 4 8 によってこれらを分割して記憶するようにする。また、送信メッセージ収集手段 1 4 5 によって、H T T Pクライアントに送信すべき動作要求とH T T Pクライアントからの動作要求に対する動作応答とを収集し、H T T Pレスポンス送信手段 1 4 6 によってこれらを一括して送信するようにする。そして、H T T Pクライアントからの動作要求があった場合には、クライアントコマンド実行結果生成手段 1 4 4 がその動作要求に係る動作を実行して動作応答を生成する。

**【選択図】** 図 2 2



特願 2 0 0 3 - 3 0 5 5 1 1

出 願 人 履 歴 情 報

識別番号

[ 0 0 0 0 0 6 7 4 7 ]

1. 変更年月日

2 0 0 2 年 5 月 1 7 日

[変更理由]

住所変更

住 所

東京都大田区中馬込 1 丁目 3 番 6 号

氏 名

株式会社リコー